НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

До захисту допущено:

Завідувач кафедри

_____ Олександр РОЛІК

«___»_____20__ р.

Дипломний проєкт

на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інформаційні управляючі системи та технології»
спеціальності 126 «Інформаційні системи та технології»
на тему: «Телеграм чат бот-словник»

Виконав:
студент IV курсу, групи ІК-91
Кузьмук Ярослав Романович _____

Керівник:
доцент, доктор технічних наук,
Поліщук Михайло Михайлович _____

Рецензент:
Посада, науковий ступінь, вчене звання,
Прізвище, ім’я, по батькові _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2023 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційне забезпечення робототехнічних систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____20__ р.

ЗАВДАННЯ
на дипломний проєкт студенту
Кузьмуку Ярославу Романовичу

1. Тема проєкту « Телеграм чат бот-словник», керівник проєкту Поліщук Михайло Михайлович, доцент, кандидат технічних наук, затверджені наказом по університету від «31» травня 2023 р. No2101-с

2. Термін подання студентом проєкту: «12» червня 2023 року;

3. Вихідні дані до проєкту: Технічне завдання;

4. Зміст пояснювальної записки:

1. Загальні положення: основні визначення та терміни, опис предметного середовища, огляд наявних аналогів, постановка задачі;

2. Інформаційне забезпечення: вхідні та вихідні дані, опис структури бази даних;

3. Програмне та технічне забезпечення: засоби та інструменти розробки, вимоги до технічного забезпечення, архітектура програмного забезпечення;

4. Технологічний розділ: керівництво користувача, опис випробовування програмного продукту.

5. Перелік графічного матеріалу (із зазначенням обов’язкових креслеників, плакатів, презентацій тощо)

1.Діаграма послідовності (А3)

2. Діаграма класів (А3)

3. Структура бази даних та схема використання застосунку (А3)

4. Схеми рівневої архітектури та структури компонентів (А3)

6. Дата видачі завдання 27 лютого 2023 року.

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	Аналіз предметної області	18.04.2023	
2	Огляд та аналіз існуючих рішень	22.04.2023	
3	Опис постановки задачі	27.04.2023	
4	Створення архітектури технічного рішення	30.04.2023	
5	Розробка та опис структури бази даних	04.05.2023	
6	Розробка та опис технічного рішення	08.05.2023	
7	Тестування проєкту	27.05.2023	
8	Робота над помилками, зауваженнями	29.05.2023	
9	Написання документації	01.06.2023	

Студент Ярослав КУЗЬМУК

Керівник Михайло ПОЛІЩУК

АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка дипломного проєкту складається с п’яти розділів, містить __ рисунків, __ таблиць, __ додатків, __ джерел.

У сучасному світі, де ми маємо безліч інформації та можливостей, володіння мовами є одним із ключових навичок. Для багатьох людей вивчення англійської мови є важливим завданням, але часто виникає проблема зі збереженням та повторенням вивчених слів. Для вирішення цієї проблеми та надання зручного інструменту для вивчення слів на англійській мові був розроблений телеграм чат бот-словник.

Даний дипломний проєкт присвячений дослідженню існуючих програм, які допомагають з вивченням англійської мови, а також створення власного застосунку з функціями збереження, перегляду слів, проходження тестів з цими словами та можливості слідкування за власним прогресом за допомогою графіків статистики.

Метою цього проекту є створення інтерактивного словника, який дозволяє користувачам зберігати слова на англійській мові та отримувати їх переклад на українську. Також, бот надає можливість проходження тестів для перевірки знань користувача. Різні рівні тестів дозволяють пройти перевірку з розширенням лексичного запасу та практикуванням різних навичок, пов'язаних з вивченням англійської мови.

Ключові слова: - ТЕЛЕГРАМ БОТ, ВИВЧЕННЯ АНГЛІЙСЬКОЇ МОВИ.

ABSTRACT

In today's world, where we have a lot of information and opportunities, language skills are one of the key skills. For many people, learning English is an important task, but there is often a problem with retaining and repeating the learned words. Telegram chat bot dictionary was developed to solve this problem and provide a convenient tool for learning English words.

This diploma project is dedicated to the research of existing programs that help with learning English, as well as the creation of your own application with the functions of saving, viewing words, passing tests with these words and the ability to follow your own progress with the help of statistics graphs.

The goal of this project is to create an interactive dictionary that allows users to store words in English and receive their translation into Ukrainian. Also, the bot provides an opportunity to pass tests to check the user's knowledge. Different levels of tests allow you to pass the test with the expansion of vocabulary and practice of various skills related to the study of the English language.

Keywords: - TELEGRAM BOT, LEARNING ENGLISH.

Номер рядка	Формат	Позначення	Найменування	Кільк. аркушів	Номер екзем.	Примітка
1			Документація загальна			
2						
3			Знову розроблена			
4						
5	А4	ІК91.270БАК.005 ПЗ	Пояснювальна записка	60		
6	А3	ІК91.270БАК.005 Д1	Телеграм чат бот-словник.	1		
7			Діаграма послідовності			
8	А3	ІК91.270БАК.005 Д2	Телеграм чат бот-словник.	1		
9			Діаграма класів			
10	А3	ІК91.270БАК.005 Д3	Телеграм чат бот-словник.	1		
11			Структура бази даних та схема			
12			використання застосунку			
13	А3	ІК91.270БАК.005 Д4	Телеграм чат бот словник.	1		
14			Схеми рівневої архітектури та			
15			структури компонентів			
16						
17						
18						
19						
20						
21						
22						
23						
24						
25						
26						
27						
28						

Зм.	Аркуш	№ докум.	Підпис	Дата	ІК91.270БАК.005 ТП		
Розроб.		Кузьмук Я. Р.			Телеграм чат бот-словник. Відомість проєкту	Літ.	Аркуш 1 / Аркушів 1
Керівн.		Поліщук М. М.				Т	
						КПІ ім. Ігоря Сікорського Група ІК-91	
Затв.							

Пояснювальна записка
до дипломного проєкту
на тему: «Телеграм чат бот-словник»

Київ – 2023 року

ЗМІСТ

ВСТУП

Додатки для вивчення англійської мови виконують важливу роль у сучасному навчанні та покращенні мовних навичок. Вони стали популярним інструментом, оскільки пропонують користувачам інтерактивне, доступне і ефективне середовище для вивчення мови.

Доступність і зручність, інтерактивність та адаптація до потреб користувача, велика кількість вправ і ресурсів, миттєвий зворотній зв’язок. Всі ці аспекти роблять додатки для вивчення англійської мови ефективними і популярними інструментами для самостійного навчання, покращення мовних навичок та підтримки мовного прогресу. Вони створюють динамічне і стимулююче середовище, яке допомагає користувачам досягти своїх мовних цілей.

Під час створення застосунку для вивчення мови, треба визначити основні його елементи та функції, які зроблять процес вивчення цікавим та ефективним для користувача. Можливість мати власний словник зробить додаток гнучким, і користувач буде мати можливість самостійно визначати релевантність тих чи інших слів. Але просто мати список слів – не дуже ефективно, застосунок повинен впроваджувати інтерактив. Для цього ідеально підходять тести, користувач буде мати можливість перевірити знання слів з власного словника. Також важливим елементом є можливість слідкувати за прогресом, для цього підходять графіки статистики проходження тестів.

Метою дипломного проєкту є створення телеграм бота, для вивчення англійської мови, з функціями маніпуляцій з власним словником, проходженням тестів зі доданими словами та переглядом статистики прогресу.

Для досягнення необхідної мети треба реалізувати комплекс наступних задач:

– розробка архітектури застосунку;

– розробка бази даних;

– розробка верхорівневих модулів для роботи зі словами, статистикою та тестами;

– розробка телеграм модулю в якості інтерфейсу для користувача.

Практичне значення одержаних результатів. Створений застосунок допоможе користувачам з вивченням англійської мови, в якості додаткового джерела навчання.

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Опис предметного середовища

На сьогоднішній день традиційне вивчення англійської мови по підручникам відходить на другий план. Перевагу беруть такі засоби як: споживання контенту на англійській мові, спілкування з носіями мови та додатки. Існує декілька типів застосунків, які допомагають вивчати англійську мову.

Найпоширеніший варіант подібних додатків – застосунки, в основі яких лежить курс. Тобто користувач проходить розділи, різного ступеня складності, поповнюючи свої знання словами, лексикою та орфографією. Зазвичай такі додатки мають велику кількість матеріалів, на кшталт відео, текстів та уроків. Також невід'ємним елементом є наявність різноманітних тестів, які перевіряють різні аспекти мови.

Також є більш простий, але доволі ефективний варіант – контакт з носіями мови. Деякі додатки дозволяють навчатися на практиці, вони дають користувачу можливість зв'язку з зацікавленими в спілкуванні іноземцями. Через дзвінки або чат користувач спілкується з цими людьми, навчаючись формулювати свої думки та розуміти слова співбесідника. Таким чином поповнюється словниковий запас та здобуваються навички ведення реального діалогу.

Найбільш гнучким функціоналом, який дає користувачу можливість ставити пріоритети самому є словник. Подібні застосунки впроваджують можливість додавати слова, які інтересують конкретного користувача. Таким чином можна самому будувати власний словник, та вивчати слова в ньому. Також важливим елементом словника є тести з його словами, для перевірки знань користувача.

Таблиця 1.1 – Особливості предметного середовища застосунку для вивчення мови

Особливість	Опис
Електронні матеріали	Застосунок для вивчення мови може надавати відео, текстові матеріали та уроки для вивчення мови

Власний словник	Застосунок для вивчення мови може надавати можливість зберігати, переглядати та видаляти слова у власному словнику
Проходження тестів	Застосунок для вивчення мови надає можливість проходити тести для покращення знань
Збереження прогресу	Застосунок для вивчення мови дозволяє спостерігати за прогресом навчання
Спілкування з носіями мови	Застосунок для вивчення мови може надавати можливість спілкування з носіями мови
Переклад слів	Застосунок може перекладати потрібні користувачу слова

1.1.1 Опис процесу діяльності

Під час створення додатку-словника для вивчення англійської мови, за головний пріоритет береться не об’єм матеріалів для навчання або можливість спілкуватись з носіями мови, а саме гнучкість. Тож основна сутність системи – слово. Уся логіка зав’язана саме на можливості зберігати, видаляти та переглядати додані слова. Також логіка тестів теж зав’язана на словах.

Отже, першим етапом розробки необхідно поставити модуль слів. Саме він відповідальний за перераховані вище маніпуляції зі словами (додавання, видалення та перегляд).

Наступним кроком треба розробити логіку роботи зі статистикою. Користувачу важливо надати можливість слідкування за власним прогресом. Цей модуль буде відповідальним за генерацію саме графіків проходження тестів.

Останній та самий важливий модуль – модуль тестів. Саме він дозволяю користувачу вдосконалювати свої знання, та допомагає в запам’ятовуванні нових слів. Кожен тест буде згенерований на основі вже існуючих слів у словнику.

1.1.2 Опис функціональної моделі

Додаток-словник буде мати тільки одного актора – користувача. Розробимо схему використання застосунку.

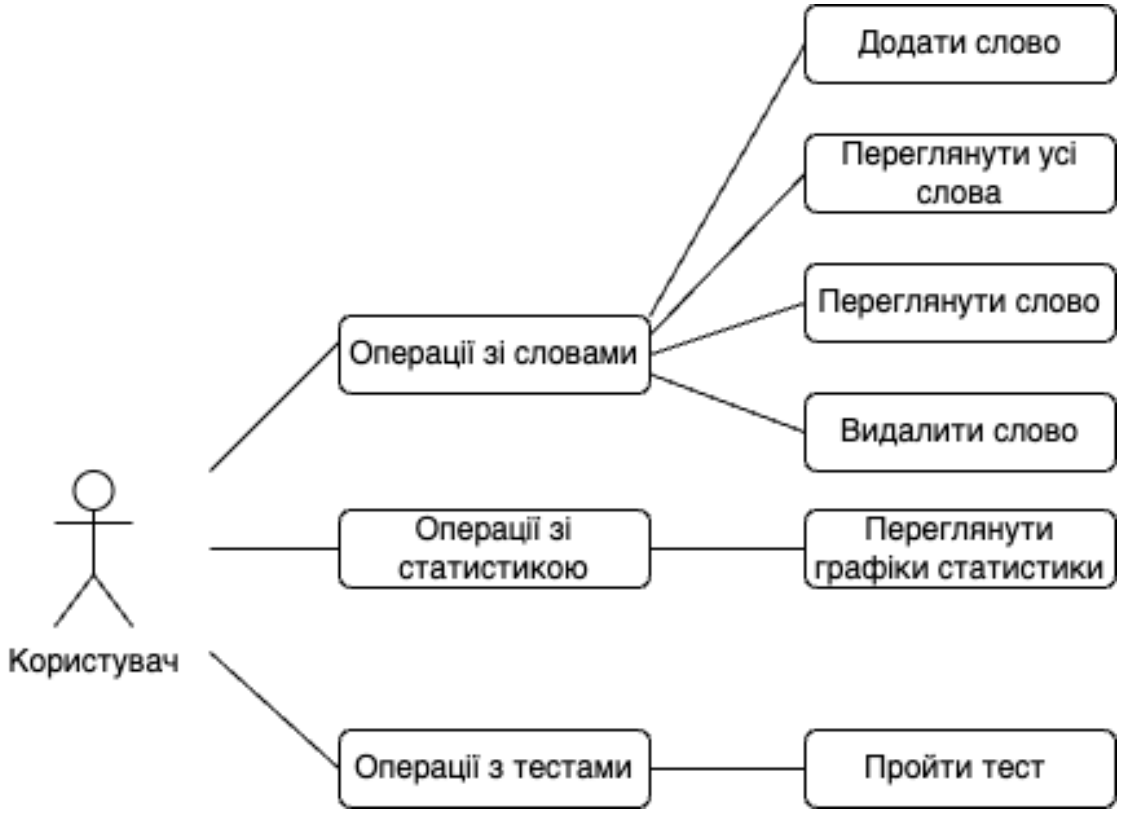

Рисунок 1.1 – Схема використання застосунку користувачем

Усі функції, які використовує користувач наведені в таблиці 1.2 з відповідними пріоритетами.

Таблиця 1.2 – Функції користувача та їх пріоритети

Функція	Опис функції	Пріоритет
Додання слова	Бот додає введене користувачем слово,	Високий

	перекладає її на українську та зберігає у словник	
Перегляд усіх слів	Бот надає користувачу список слів, які він має у своєму словнику с перекладом на українську	Високий
Перегляд слова	Бот надає слово з перекладом по введеній користувачем англійській версії цього слова	Низький
Видалення слова	Бот видаляє слово зі словника по введеній користувачем англійській версії цього слова	Низький
Перегляд графіків статистики	Бот генерує на надає користувачу графіки статистики проходження користувачем тестів по днях	Високий
Проходження тестів	Бот надає генерує для користувача тест на основі одного зі слів за словника	Високий

1.2 Огляд наявних аналогів

Перед розробкою телеграм чат бот-словника було проведено дослідження аналогічних застосунків, які використовуються для вивчення слів іноземних мов. Це дослідження було здійснене з метою визначення основних можливостей та функціоналу конкуруючих програм, а також для виявлення їх переваг та недоліків.

Для приклади були розглянуті додатки:

- Tandem
- Duolingo
- U-Dictionary

Tandem є мобільним додатком для вивчення мови, який сприяє практичному вдосконаленню мовленнєвих навичок та спілкуванню з носіями мови з усього світу. Додаток створений з метою забезпечити користувачам можливість обміну мовними навичками та взаємну підтримку у вивченні мови.

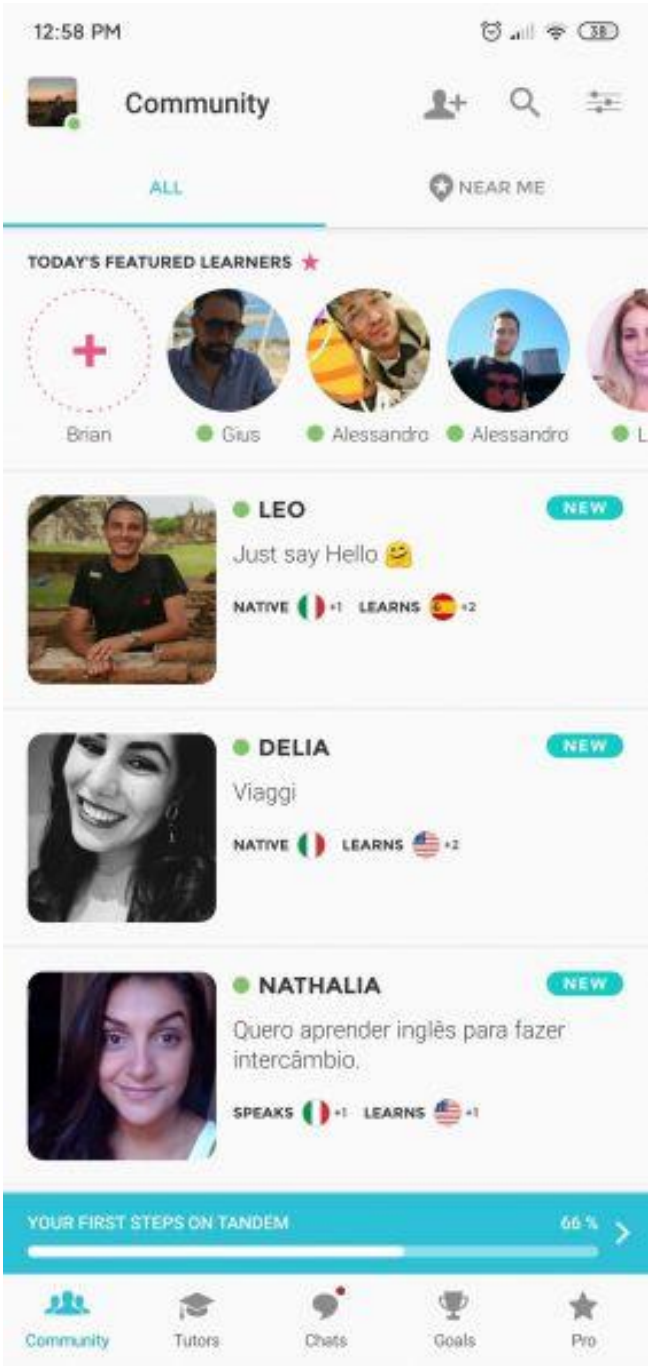

Рисунок 1.2 – Знімок інтерфейсу Tandem

Основна ідея Tandem полягає у тому, щоб знайти партнера для спілкування, який має рідну мову, яку ви хочете вивчити, і хоче вивчити вашу рідну мову. Завдяки цьому, обидва партнери можуть отримати практику в мовленні, вдосконалювати свої навички розмовної мови та отримувати корисні відгуки та поради від носіїв мови.

Основні можливості додатку Tandem включають:

– пошук партнера: Додаток дозволяє знайти партнера для спілкування, який відповідає вашим мовним потребам та інтересам;

– обмін повідомленнями: Ви можете спілкуватися з вашим партнером через текстові повідомлення, щоб практикувати письмовий спілкування та отримувати корекцію ваших текстів;

– голосові та відео-дзвінки: Додаток дозволяє здійснювати голосові та відео-дзвінки з вашим партнером, щоб практикувати усну мову та поліпшувати вимову;

– корисні матеріали: Tandem надає доступ до різноманітних матеріалів для вивчення мови, таких як статті, вправи, аудіо- та відеоматеріали;

– культурний обмін: Ви можете ділитися своїм культурним досвідом та вивчати культуру вашого партнера, що додає цікавості до спілкування;

– відгуки та оцінки: Ви можете залишати відгуки та оцінки своїм партнерам за їх допомогу та прогрес у вивченні мови.

Tandem допомагає створити позитивне та стимулююче середовище для вивчення мови, де ви можете зустріти нових людей, практикувати мовлення та підвищувати свої мовні навички.

Duolingo є одним з найпопулярніших мобільних додатків для вивчення мов. Він пропонує інтерактивну та групову форму вивчення мови, що робить процес навчання цікавим та захоплюючим.

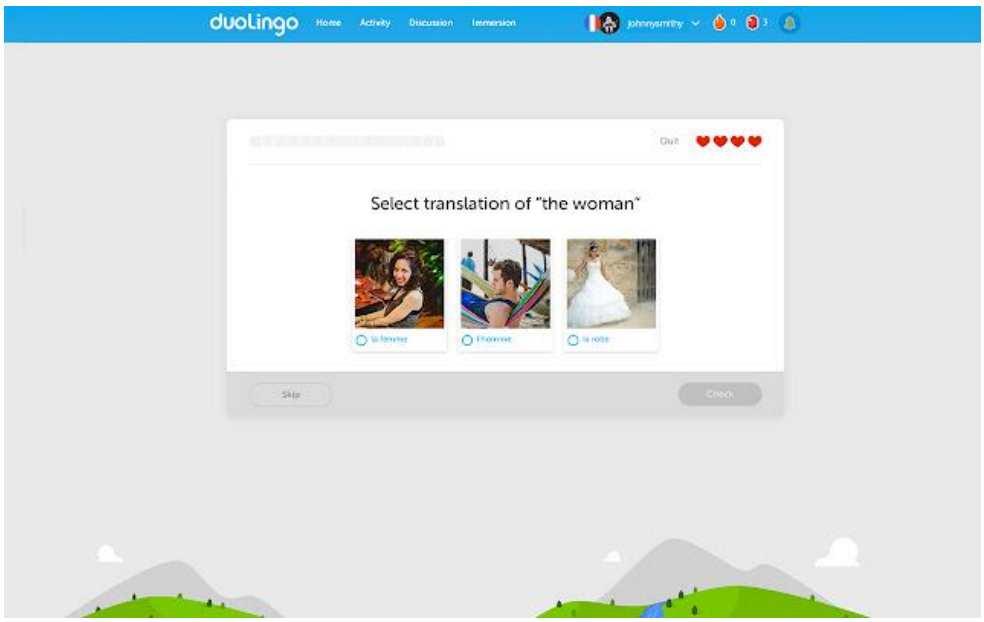

Рисунок 1.3 – Знімок інтерфейсу Duolingo

Основна мета Duolingo полягає в тому, щоб зробити вивчення мови легким, доступним та ефективним для всіх користувачів. Додаток пропонує широкий вибір мов, включаючи англійську, іспанську, французьку, німецьку, італійську, японську та багато інших.

Основні особливості та можливості додатка Duolingo включають:

– уроки та вправи: Duolingo пропонує структуровані уроки, які охоплюють всі аспекти мови, включаючи лексику, граматику, читання, письмо та вимову. Уроки включають різноманітні вправи, такі як переклад, вибір правильної відповіді, співставлення слів та багато інших;

– геймифікація: Duolingo використовує елементи гри для стимулювання навчання. Користувачі отримують очки та досягнення за виконання завдань та успішне виконання вправ. Це стимулює постійне вдосконалення та допомагає зберігати мотивацію учнів;

– персоналізований підхід: Додаток адаптується до потреб користувача та навчального темпу. Він пропонує повторення та попередні вправи для закріплення вивченого матеріалу та регулює складність уроків відповідно до рівня знань користувача;

– зручний спосіб навчання: Duolingo працює на мобільних пристроях та веб-платформі, що робить його доступним у будь-який зручний час та місце. Користувачі можуть вчитися вдома, у дорозі чи навіть у вільний час;

– спільнота користувачів: Duolingo має велику спільноту користувачів з усього світу. Користувачі можуть спілкуватися між собою, використовуючи вбудовану функцію обговорень, допомагати одне одному у вивченні мови та навіть проводити мовні практики через чат або голосові дзвінки.

Duolingo є зручним, ефективним та захоплюючим інструментом для вивчення мов. Він допомагає користувачам розвивати навички мовлення, розуміння, читання та письма через інтерактивні уроки та геймифікацію.

Додаток **U-Dictionary** є популярним мобільним словником та перекладачем, доступним для платформ Android та iOS. Він пропонує широкий спектр функцій для допомоги користувачам у вивченні та розширенні свого словникового запасу.

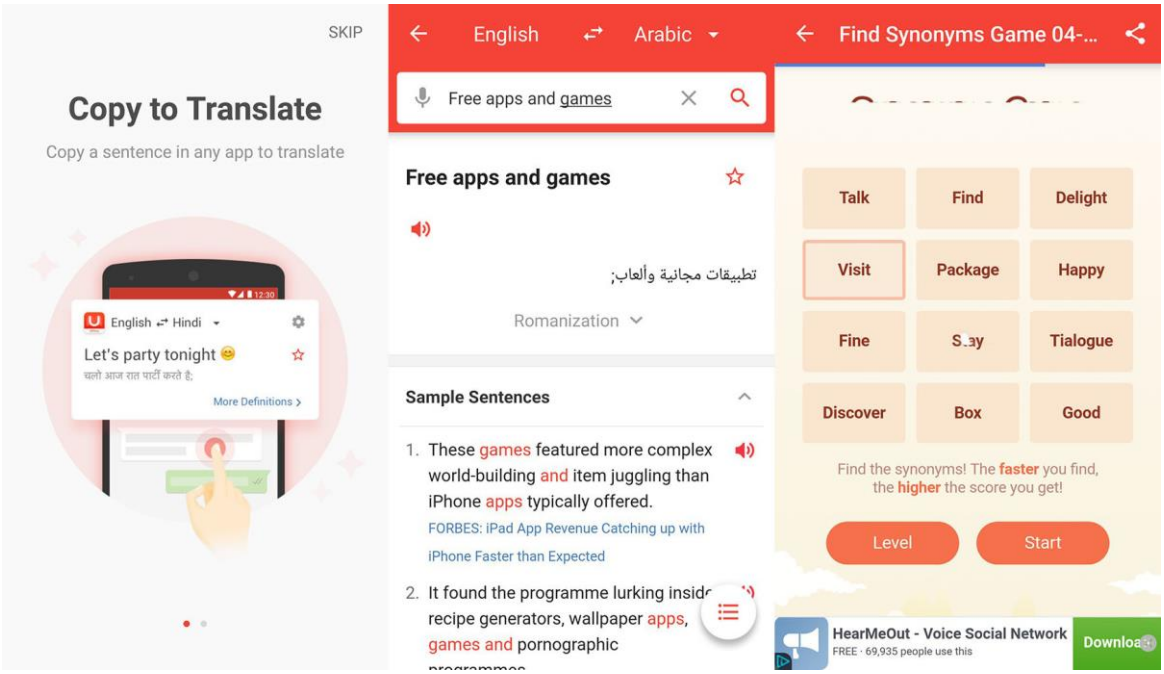

Рисунок 1.4 – Знімок інтерфейсу U-Dictionary

Ось деякі ключові особливості додатку U-Dictionary:

– багатомовний словник: U-Dictionary містить велику базу даних слів і фраз з різних мов, включаючи англійську, іспанську, французьку, німецьку, італійську, хінді та багато інших. Користувачі можуть швидко знайти переклади, визначення та приклади вживання слів у різних мовах;

– офлайн-режим: Додаток дозволяє завантажити словники для використання в офлайн-режимі. Це особливо зручно для користувачів, які подорожують або перебувають в областях з обмеженим доступом до Інтернету;

– вимова та відтворення звуку: U-Dictionary надає можливість прослуховувати вимову слів та фраз, що допомагає користувачам вивчати правильну вимову та акустичний контекст;

– камера-перекладач: Додаток має функцію камера-перекладача, яка дозволяє користувачам перекладати текст, знімаючи його з фотографій або скануючи його на екрані. Це зручно для швидкого перекладу написів, посібників або інших текстів без необхідності вручного вводу;

– віджет на екрані: U-Dictionary надає віджет, який можна розмістити на домашньому екрані мобільного пристрою. Це дозволяє швидко перекладати слова та отримувати визначення без необхідності відкривати додаток;

– слова дня та вікторини: Додаток пропонує щоденні слова дня, які допомагають розширити словниковий запас. Також є різноманітні вікторини, тести та інтерактивні завдання для практики знань;

– переклад тексту в месенджерах: U-Dictionary підтримує переклад тексту безпосередньо в різних месенджерах, таких як WhatsApp, Facebook Messenger, Viber тощо. Це зручно для комунікації зі співрозмовниками на різних мовах.

Це лише кілька особливостей додатку U-Dictionary. Загалом, він є потужним інструментом для перекладу та вивчення слів та мов, який може бути корисним для студентів, мовних ентузіастів та тих, хто хоче поліпшити свої мовні навички.

Розглянувши існуючи додатки для вивчення мови сформулюємо таблицю порівнянь аналогів.

Таблиця 1.3 – Порівняння аналогів додатків для вивчення мови

Функція	Tandem	Duolingo	U-Dictionary	Даний застосунок
Переклад конкретних слів	-	-	+	+
Матеріали для навчання	+	+	-	-
Тести на знання	-	+	+	+
Власний словник	-	-	-	+

Спілкування з носіями мови	+	-	-	-

1.3 Постановка задачі

1.3.1 Призначення розробки

Система призначена для допомоги користувачу у вивченні англійської, розширення словникового запасу, покращення розуміння використання різних слів у контексті.

1.3.2 Цілі та задачі розробки

Метою дипломного проєкту є створення застосунку, для допомоги вивчення англійської мови, з функціями маніпуляцій з власним словником, проходженням тестів з доданими словами та переглядом статистики прогресу у вигляді графіків.

Для досягнення поставленої мети необхідно:

– розробити модуль маніпуляцій зі словами у персональному словнику;

– розробити модуль створення та оновлення статистики, а також генерації графіків на її основі;

– розробити модуль генерації тестів;

– розробити модуль, який відкриє доступ до функціоналу застосунку для користувача через команди в телеграм.

Висновки до розділу

В даному розділі було спроектоване рішення додатку для вивчення англійської. Були описані предметне середовище та процес діяльності. Був описаний функціонал застосунку.

Були проаналізовані аналогічні додатки, та проведено їх порівняння.

Була визначена мета продукту та задачі для її досягнення.

Можна зробити висновок, що аналогічні додатки для вивчення мови мають широкий функціонал та привабливий інтерфейс, але також можуть відволікати користувача різноманіттю наданих функцій. Також вони мають недешеву підписку.

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Вхідні дані

Фактично усі вхідні дані є телеграм командами, які вводить користувач (4 з яких мають параметр). Також є окремий випадок, коли користувач вводить дані в якості відповіді на тест. Розберемо види даних по сутностям системи.

Слова:

- додавання слів – команда + слово англійською;
- отримання списку слів – команда;
- отримання конкретного слова – команда + слово англійською;
- видалення слова – команда + слово англійською.

Статистика:

- отримання статистики – команда.

Тести:

- отримання тесту на проходження – команда + рівень складності тесту (1, 2 чи 3);
- відповідь на тест першого рівня – вибір користувача з варіантів у «Вікторині», яку надсилає бот;
- відповідь на тести другого та третього рівнів – слово англійською.

2.2 Вихідні дані

Вихідні дані системи також розіб’ємо по сутностям.

Слова:

- додавання слів – створення нового слова в базі, сповіщення про успішне або ні додання слова у словник або список пропонованих слів, якщо введене було з помилкою;
- отримання списку слів – список слів зі словника англійською та українською мовами або сповіщення про пустий словник;

– отримання конкретного слова – слово англійською та українською мовами або сповіщення про відсутність цього слова;

– видалення слова – видалення слова з бази, сповіщення про успішне або ні видалення слова.

Статистика

– отримання статистики – 2 згенерованих зображення статистики у вигляді графіків.

Тести

– отримання тесту на проходження – «Вікторина» з трьома варіантами перекладу слова зі словника на англійську;

– відповіді на тести – сповіщення про успішно або неправильно складений тест.

Найважливіший елемент функціоналу телеграм чат бот-словника, який стосується саме допомоги користувачу в вивченні нових слів є можливість проходження тестів з вивченими словами. Тести мають три рівні складності:

– перший рівень - самий простий, на ньому бот пропонує користувачу одне зі збережених слів українською мовою та надає 3 варіанти його перекладу на англійську. Користувач повинен вибрати правильний варіант. В цьому виді тестів є нюанс, який і робить цей рівень найпростішим - усі варіанти перекладу слова беруться із вже збережених слів, тобто якщо користувач бачить знайомий варіант серед відповідей, йому простіше вибрати правильну. Також такий вид тестів не доречно використовувати коли в базі збережено маленька кількість слів. Якщо користувач відповів правильно - йому нараховується 1 бал;

– другий рівень - трохи складніший за перший, на ньому користувачу теж пропонується одне зі збережених слів українською мовою, але тепер замість варіантів відповіді користувач повинен самостійно написати переклад на англійську. Такі тести перевіряють не тільки словниковий запас користувача, а й знання граматики. На цьому рівні нараховується 2 бали за правильну відповідь;

– третій рівень - самий складний, на ньому перевіряється не тільки словниковий запас та граматика, але й здатність користувача розуміти використання конкретного слова в якомусь контексті. Бот надає випадкове речення

англійською, в якому є одне зі збережених слів, замість якого в реченні вставляється пропуск (речення беруться з сайту [1]), користувач в свою чергу повинен написати яке саме слово повинно стояти на місці пробілу. На цьому рівні нараховується 3 бали за правильну відповідь.

Для покращення навчального процесу бот не дає можливості пройти тест з одним і тим самим словом кілька раз на день. Тобто гарантовано всі тести пройдені за день будуть з різними словами (крім випадку, коли користувач вже застосував всі слова для тестів на день, тоді будуть повтори).

2.3 Опис структури бази даних

На рисунку 2.1 наведена структура бази даних

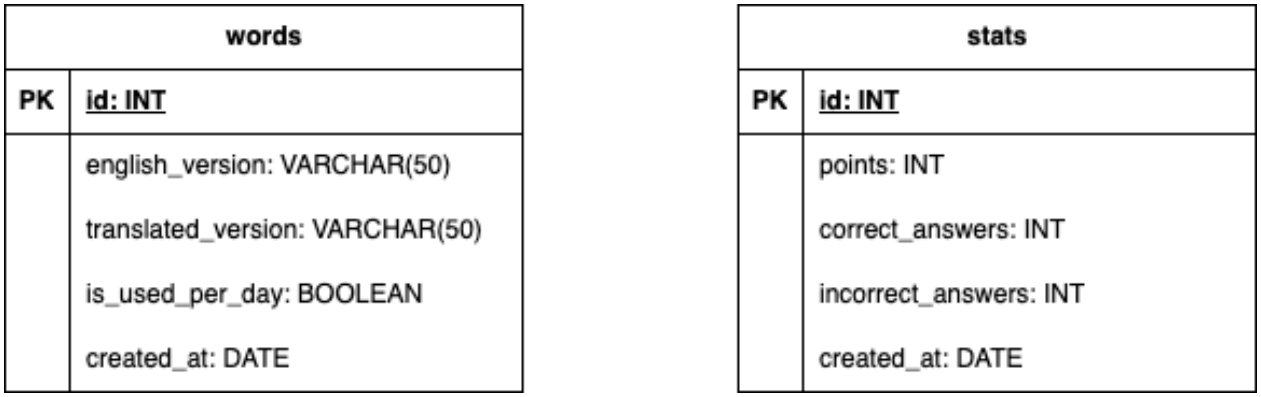

Рисунок 2.1 – Структурна схема бази даних

Таблиця 2.1 – Опис таблиць бази даних

Таблиця	Призначення
words	Таблиця для збереження слів користувача
stats	Таблиця для збереження статистики проходження тестів користувачем

Таблиця 2.2 – Опис полів таблиць у базі даних

Таблиця	Поле	Опис
words	id	Первинний ключ таблиці
	english_version	Англійська версія слова
	translated_version	Українська версія слова
	is_used_per_day	Індикатор, якій вказує, чи було слово використано за день
	created_at	Час додання слова
stats	id	Первинний ключ таблиці
	points	Кількість зарахованих балів за тести за один день
	correct_answers	Кількість правильно пройдених тестів за день
	incorrect_answers	Кількість неправильно пройдених тестів за день
	created_at	Дата, якій відповідає конкретна статистика

Висновки до розділу

В даному розділі була розроблена структура бази даних, створений опис таблиць та їх полів. Для повноцінного функціонування застосунку достатньо лише двох таблиць без зав’язків між ними, тому що всі інші дані створюються під час роботи додатку, та одразу пропадають після використання. Створення коректної структури бази даних є надзвичайно важливим етапом при

розробці програмного забезпечення, оскільки вона визначає основу для ефективного та надійного зберігання, організації та доступу до даних. Структуру бази даних важливо проєктувати таким чином, щоб усі важливі дані не пропадали, а зайві – не зберігались та не займали у базі простору.

3 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Засоби розробки

Для розробки телеграм чат бот-словника було використано Node.js та TypeScript. Node.js є платформою, яка дозволяє розробляти серверні застосунки з використанням JavaScript. TypeScript - це розширення JavaScript, яке надає типізацію та інші покращення для розробки програмного забезпечення. Використання цих технологій дозволяє створити потужну та масштабовану систему.

Node.js є відкритим середовищем виконання JavaScript, побудованим на базі движка V8, який розробляється компанією Google. Він дозволяє виконувати JavaScript-код на серверній стороні і побудовувати різноманітні додатки та сервіси.

Основні особливості Node.js включають:

– потокова архітектура: Node.js побудований на принципі однопотокового, подієвого виконання. Це означає, що він здатний обробляти багато запитів одночасно, не блокуючи виконання інших операцій. Це робить його особливо підходящим для розробки високонавантажених додатків, таких як веб-сервери;

– широкі можливості: Node.js має велику кількість модулів та пакетів, доступних через систему керування пакетами npm. Це дозволяє розробникам використовувати готові рішення для різних завдань, таких як обробка HTTP-запитів, робота з базами даних, робота з файловою системою, робота з мережевими протоколами тощо;

– висока продуктивність: Node.js використовує двигун V8, який забезпечує швидке виконання JavaScript-коду. Це дозволяє розробникам створювати ефективні та продуктивні додатки, забезпечуючи високу швидкодію і низький рівень завантаження ресурсів;

– складаність мережевих додатків: Node.js надає потужні засоби для розробки мережевих додатків, таких як веб-сервери, веб-додатки, API і багато інших. Завдяки вбудованій бібліотеці HTTP, Node.js дозволяє легко створювати і обробляти HTTP-запити та відповіді;

– розширення можливостей: Node.js підтримує використання нативних модулів, що дозволяє розробникам використовувати функціональність, доступну на рівні операційної системи, таку як робота з файлами, мережею, криптографією тощо. Це дозволяє розширити можливості Node.js і використовувати його для різноманітних завдань.

Node.js став популярним серед розробників завдяки своїй простоті, продуктивності і широким можливостям. Він застосовується для розробки серверних додатків, веб-додатків, мікросервісів, інструментів командного рядка та багатьох інших застосувань.

TypeScript є мовою програмування, яка є розширенням мови JavaScript. Вона надає додаткові можливості для створення складних і масштабованих додатків, дозволяючи використовувати статичний типізацію і об'єктно-орієнтовані підходи.

Основні особливості TypeScript:

– статична типізація: TypeScript дозволяє визначати типи змінних, параметрів функцій, повертаємого значення функцій та інших елементів програми. Це дозволяє виявляти помилки на етапі компіляції і полегшує розробку, підтримку і рефакторинг коду;

– об'єктно-орієнтований підхід: TypeScript підтримує класи, спадкування, інтерфейси, модифікатори доступу та інші об'єктно-орієнтовані концепції, які допомагають структурувати код і покращують його перевикористовуваність;

– підтримка сучасних функцій JavaScript: TypeScript підтримує всі можливості сучасного стандарту ECMAScript, включаючи стрілкові функції, розширену роботу з промісами, async/await та інші функції, що полегшують асинхронне програмування;

– інструменти розробки: TypeScript має велику кількість інструментів, що допомагають в розробці, таких як компілятор TypeScript (tsc), інтеграція з редакторами коду, системи автодоповнень, відладчики та інші;

– підтримка існуючого коду JavaScript: Однією з ключових особливостей TypeScript є його здатність працювати з існуючим JavaScript-кодом без змін. Це

дозволяє поступово переходити до використання TypeScript, додавати типи поступово і покращувати код без необхідності повного переписування програми.

TypeScript допомагає зробити розробку JavaScript-додатків більш безпечною, підтримуваною та швидкою, забезпечуючи більшу контрольованість над програмним кодом і полегшуючи спільну роботу над проектами.

PostgreSQL - це потужна об'єктно-реляційна система управління базами даних (СУБД), яка надає широкі можливості для зберігання, організації та керування даними. Основні особливості PostgreSQL включають:

– надійність: PostgreSQL відомий своєю надійністю та стабільністю. Він підтримує транзакції з гарантованою цілісністю даних, механізми відновлення після відмови та резервне копіювання для забезпечення надійності даних;

– розширюваність: PostgreSQL має гнучку архітектуру, яка дозволяє розширювати його функціональність за допомогою розширень і власних типів даних. Це дозволяє пристосовувати систему до специфічних потреб проекту;

– підтримка SQL: PostgreSQL повністю підтримує мову структурованих запитів SQL, включаючи складні запити, підзапити, агрегаційні функції та інші розширені можливості. Він також підтримує індекси, внешні ключі, тригери та інші засоби для забезпечення ефективного доступу до даних;

– підтримка розподіленого обчислення: PostgreSQL може працювати в режимі розподіленого обчислення, що дозволяє обробляти великі обсяги даних на кластері серверів. Він підтримує реплікацію, федеровані таблиці та інші механізми для розподіленого зберігання та обробки даних;

– підтримка різних програмних мов: PostgreSQL надає можливість розробляти функції та процедури бази даних на різних мовах програмування, таких як SQL, PL/pgSQL, PL/Python, PL/Perl, PL/Java та інші. Це дозволяє використовувати потужність зовнішніх мов для реалізації складних обчислень в межах бази даних;

– велике співтовариство та підтримка: PostgreSQL має активне співтовариство розробників та користувачів, яке забезпечує підтримку, документацію, форуми обговорень та інші ресурси для допомоги користувачам.

PostgreSQL є відкритим програмним забезпеченням і доступний безкоштовно. Він використовується у багатьох веб-додатках, корпоративних системах та наукових дослідженнях як надійна та потужна СУБД.

Для отримання випадкового речення, яке містить конкретне слово було обрано рішення діставати речення з сайту [1], який містить величезну кількість різноманітних речень з різними словами. Для взаємодії з цим сайтом був обраний інструмент **puppeteer**.

Puppeteer - це високорівнева бібліотека для Node.js, розроблена командою Chrome DevTools в Google. Вона надає можливість контролювати та автоматизувати браузер Google Chrome або Chromium за допомогою програмного коду. Puppeteer надає потужний API, який дозволяє здійснювати різноманітні дії в браузері, такі як навігація по сторінкам, заповнення форм, скролінг, взаємодія з елементами сторінки, зняття скріншотів, генерація файлів PDF та багато іншого.

Puppeteer дозволяє робити веб-скрапінг (web scraping) - отримувати дані з веб-сторінок шляхом їх автоматичного аналізу та вилучення. Ви можете здійснювати пошук та отримувати вміст певних елементів, аналізувати структуру HTML, зчитувати атрибути, текст або дані з форм, і багато іншого.

Одним з потужних функціональних можливостей Puppeteer є його здатність взаємодіяти зі сторонніми веб-сайтами. Ви можете відправляти запити на сервери, отримувати відповіді, обробляти куки, робити аутентифікацію та інші HTTP-операції.

Puppeteer дозволяє отримати доступ до багатьох характеристик браузера, таких як заголовки запитів, код статусу, зміна розміру вікна, зміна геолокації і багато іншого. Ви також можете взаємодіяти зі сторонніми JavaScript-файлами, маніпулювати cookie, кешем, обробляти події браузера, включати/вимикати JavaScript та виконувати власний JavaScript-код на сторінці.

Puppeteer дозволяє зручно керувати браузером та автоматизувати веб-додатки або збирати дані з веб-сторінок.

Для будування графіків статистики був обраний інструмент **chartjs-to-image**.

chartjs-to-image - це бібліотека або плагін, який надає можливість генерувати зображення на основі графіків, створених з використанням бібліотеки Chart.js. Зазвичай, Chart.js використовується для створення інтерактивних графіків та діаграм на веб-сторінках з використанням JavaScript. Однак, інколи може бути необхідно отримати статичне зображення графіка для подальшого використання у документах, презентаціях або інших медіа-форматах.

chartjs-to-image дозволяє експортувати графік, створений за допомогою Chart.js, у вигляді зображення в різних форматах, таких як PNG або JPEG. Він надає простий та зручний спосіб конвертувати графіки Chart.js у статичні зображення без необхідності використовувати скріншоти або інші складні методи.

Завдяки chartjs-to-image, розробники можуть програмно створювати зображення графіків на основі даних і налаштувань Chart.js. Це дозволяє автоматизувати процес генерації зображень графіків та використовувати їх в різних сценаріях, включаючи автоматичну генерацію звітів, створення графічних представлень даних і інше.

Загалом, chartjs-to-image є корисним інструментом для роботи з бібліотекою Chart.js, який допомагає створювати статичні зображення графіків з легкістю та ефективністю.

Для автоматичного перекладу збережених слів був обраний інструмент **@iamtraction/google-translate**.

@iamtraction/google-translate - це пакет для Node.js, який надає простий спосіб використання сервісу машинного перекладу Google Translate у ваших програмах. З його допомогою ви можете легко перекладати тексти або фрази з однієї мови на іншу, використовуючи потужні механізми перекладу, які надає Google.

Цей пакет забезпечує зручний інтерфейс для взаємодії з сервісом Google Translate. Ви можете використовувати його для автоматичного перекладу текстів у своїх програмах, створення мультиязичних додатків, обробки великого обсягу тексту для перекладу та багато іншого.

Інструмент @iamtraction/google-translate дозволяє вам визначати початкову мову тексту і цільову мову, на яку ви хочете його перекласти. Він підтримує

широкий спектр мов, включаючи популярні мови, такі як англійська, іспанська, французька, німецька, українська та інші. Ви також можете використовувати його для визначення мови тексту автоматично, якщо ви не знаєте точної мови вхідного тексту.

Основними функціями @iamtraction/google-translate є переклад тексту з однієї мови на іншу, включаючи переклад одного слова або цілого речення, а також отримання детальної інформації про розпізнану мову тексту. Ви також можете використовувати його для перекладу списків слів або текстових файлів.

Цей інструмент дозволяє вам легко інтегрувати функціональність перекладу Google у ваші програми на Node.js і реалізувати потрібні перекладацькі сценарії зручним способом.

Для інтегрування усього функціоналу у телеграмм було обрано інструмент **telegraf**.

Telegraf - це фреймворк для розробки ботів у месенджері Telegram з використанням мови програмування JavaScript або TypeScript. Він надає зручний і потужний спосіб створення і взаємодії з чат-ботами у Telegram.

За допомогою Telegraf ви можете легко реалізувати функціональність бота, включаючи обробку вхідних повідомлень, команд, клавіатури, подій і багато іншого. Фреймворк надає широкі можливості для створення різноманітних ботів, починаючи від простих текстових відповідей до складних інтерактивних сценаріїв.

Основні особливості Telegraf включають:

– обробка повідомлень: Telegraf дозволяє визначати обробники для різних типів повідомлень, таких як текстові повідомлення, зображення, відео, аудіо, документи тощо. Ви можете виконувати різні дії з отриманими повідомленнями, включаючи відповіді, збереження даних, взаємодію з базою даних та інше;

– команди: Telegraf дозволяє визначати команди, які можуть активуватись певними ключовими словами, наприклад, /start, /help, /settings тощо. Ви можете настроїти функціонал команд для відповідного відповідного обробника;

– клавіатури: Telegraf підтримує створення інтерактивних клавіатур для ботів. Ви можете створювати кнопки, меню і спеціальні елементи управління, які дозволяють користувачам взаємодіяти з ботом шляхом вибору певних опцій;

– міжнародизація: Telegraf підтримує різні мови та локалізацію. Ви можете настроїти бота для роботи з різними мовами та надати переклади повідомлень для різних локалізацій;

– middleware: Telegraf дозволяє використовувати middleware - проміжний програмний рівень для обробки повідомлень. Ви можете використовувати middleware для виконання певних операцій перед або після обробки повідомлень, наприклад, для логування, аутентифікації, обробки помилок тощо.

Telegraf є потужним інструментом для розробки ботів у Telegram з великим набором функціональності і простим API. Він дозволяє легко створювати і керувати ботами, які забезпечують багатогранну взаємодію з користувачами.

Для систематичного створення нової статистики у базі через рівні проміжки часу був обраний інструмент **node-cron**.

node-cron є бібліотекою для планування і виконання повторюваних завдань в середовищі Node.js. Цей інструмент дозволяє створювати розклади виконання завдань за заданими часовими інтервалами або у визначені моменти часу.

Основні особливості і можливості node-cron:

– простий синтаксис: Інтерфейс node-cron є досить простим і легким у використанні. Він дозволяє задавати часові правила за допомогою поняття "cron expressions", які базуються на стандартному синтаксисі cron, використовуваному в Unix-подібних операційних системах;

– гнучкість: node-cron дозволяє створювати розклади виконання завдань з різними часовими інтервалами. Ви можете задавати виконання завдання кожну хвилину, годину, день тижня або визначену дату і час;

– підтримка функцій зворотного виклику: Завдання, які ви плануєте з node-cron, можуть бути функціями зворотного виклику, які виконуються у вказаний момент часу. Це дозволяє вам виконувати будь-який код або функціональність, яку ви потребуєте;

– переваги асинхронного виконання: node-cron підтримує асинхронні функції зворотного виклику, що дозволяє виконувати завдання, які працюють з асинхронними операціями, такими як запити до бази даних або зовнішні API;

– керування тайм-зонами: Ви можете налаштувати виконання завдань в певній тайм-зоні, що дозволяє враховувати різницю в часі між різними регіонами.

Node-cron є потужним інструментом для планування та автоматизації повторюваних завдань в середовищі Node.js. Він може бути використаний для різних сценаріїв, таких як резервне копіювання даних, автоматичне виконання операцій або регулярне оновлення даних.

Для перевірки правильності написання введених користувачем нових слів був використаний інструмент **typo-js**.

Typo.js є інструментом для виявлення і корекції орфографічних помилок в тексті веб-додатків або в інших проектах, що використовують JavaScript. Він дозволяє автоматично виправляти орфографічні помилки, що полегшує процес редагування і покращує якість написаного тексту.

Основні особливості і можливості Typo.js:

– виявлення орфографічних помилок: Typo.js використовує вбудований словник, який містить правильні слова і їхні варіанти написання. За допомогою алгоритмів перевірки, Typo.js знаходить слова, які не зустрічаються у словнику або мають неправильний напис;

– автоматична корекція помилок: Після виявлення орфографічних помилок, Typo.js може автоматично замінити неправильно написані слова на правильні. Це полегшує процес редагування тексту та забезпечує однорідність і правильність використання слів у веб-додатках;

– розширення словника: Typo.js надає можливість розширити словник власними словами або спеціалізованими термінами, що не входять до вбудованого словника. Це дозволяє забезпечити точність перевірки орфографії для конкретних вимог проекту;

– підтримка мов: Typo.js підтримує багатомовну перевірку орфографії і може працювати з різними мовами, такими як англійська, французька, іспанська,

німецька та багато інших. Це робить його універсальним інструментом для перевірки орфографії у різних мовних середовищах.

Typo.js допомагає полегшити процес редагування тексту, забезпечує правильність написання і забезпечує більш професійний вигляд текстових веб-додатків та проектів, що використовують JavaScript.

Для взаємодії з базою даних було використано інструмент **pg**.

Інструмент pg (pg-promise) є бібліотекою для взаємодії з базою даних PostgreSQL в середовищі Node.js. Він надає зручні інтерфейси та функціонал для роботи з базою даних, дозволяючи розробникам ефективно виконувати запити, маніпулювати даними та управляти транзакціями.

Основні особливості інструменту pg:

– простота використання: pg надає простий та зрозумілий API, що дозволяє легко виконувати запити до бази даних. Він підтримує використання SQL-запитів з параметрами та підготовленими запитами, що сприяє попередженню SQL-ін'єкцій та покращує безпеку додатка;

– підтримка транзакцій: pg дозволяє працювати з транзакціями в базі даних PostgreSQL. Ви можете почати, фіксувати або відкатувати транзакції, що дозволяє забезпечити консистентність та надійність даних;

– підтримка обробки помилок: Інструмент pg має механізм обробки помилок, який дозволяє ловити та обробляти винятки, що виникають під час виконання запитів. Це допомагає забезпечити стабільність та надійність додатка;

– підтримка пулу з'єднань: pg може створювати та керувати пулом з'єднань з базою даних, що дозволяє ефективно використовувати ресурси та керувати з'єднаннями з базою даних;

– підтримка розширень: pg дозволяє використовувати розширення PostgreSQL, такі як увімкненість JSON-даних, роботу з геоданими тощо.

Інструмент pg є популярним вибором для взаємодії з базою даних PostgreSQL в середовищі Node.js, оскільки він поєднує простоту використання з потужним функціоналом та надійною продуктивністю.

3.2 Вимоги до технічного забезпечення

Даний застосунок по суті складається з серверної частини та бази даних, варто визначитись з конфігурацією технічного забезпечення для коректної роботи серверу на локальній машині.

- процесор з тактовою частотою не нижче 1.6 ГГц;
- об’єм оперативної пам’яті не менше за 4 ГБ;
- операційна система Windows / MacOS;
- Node.js вище версії 18.0.0;
- PostgreSQL 10 і вище.

3.3 Архітектура програмного забезпечення

Архітектура даного застосунку поділяється на 3 рівні, описані нижче.

Перший рівень – рівень представлення. Він містить інтерфейс, з яким взаємодіє користувач. Фактично – це рівень вводу / виводу, він передає введені користувачем дані, відправляє їх на обробку до високорівневої бізнес логіки, та виводить результат. Також цей рівень відповідальний за валідацію даних. Єдина відмінність від класичного рівню представлення, уся вище описана логіка знаходиться не в окремому клієнтському сервісі, а є одним з модулів серверної частини, реалізованим з допомогою бібліотеки Telegraf (тому що інтерфейс користувача у варіанті даного застосунку – це телеграм бот).

Другий рівень поділяється на два елемента – рівень бізнес-логіки та рівень доступу до даних. Рівень бізнес-логіки відповідальний за різноманітні маніпуляції з даними переданими рівнем представлення. Бізнес-логіка представлена набором класів та функцій, які викликаються безпосередньо у модулі представлення.

Рівень доступу до даних є набором функцій, які представляють собою запити до бази даних. Він зроблений, для відокремлення важливої бізнес-логіки від маніпуляцій з базою, роблячи бізнес-логіку незалежною від процесу будування комунікації з базою даних.

Третій рівень – рівень сховища даних. Цей рівень представлений PostgreSQL базою даних, в якій зберігаються інформація про статистику та збережені користувачем слова.

На рисунку 3.1 наведена діаграма структури додатку.

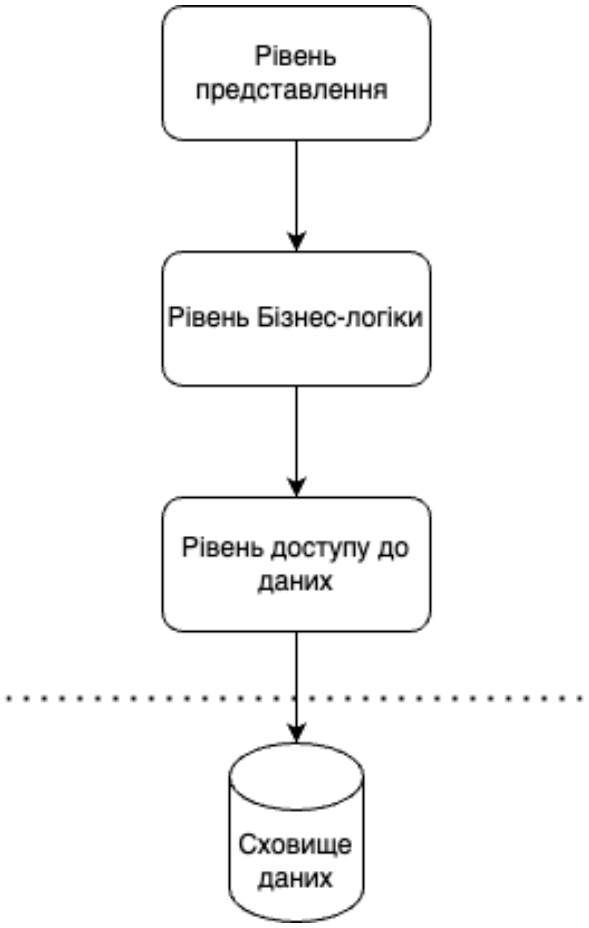

Рисунок 3.1 – Діаграма структури додатку

Таким чином окремі блоки системи ізольовані один від одного , що позволяє вносити редагування в окремі модулі, без шкоди для інших. Найважливішим моментом є створення рівню бізнес-логіки незалежним. При необхідності можна без проблем переробити рівень доступу до даних із використанням іншого інструменту, або додати контролери та винести рівень представлення в окремий клієнтський сервіс.

Отже, структура, не зважаючи на її простоту, залишається гнучкою та зручною, в плані масштабування системи.

3.3.1 Діаграма класів

Реалізація даного додатку складається з п’яти основних сервіс-класів, одного допоміжного класу, та класів статусів. Сервіс-класи будуть розглянуті в діаграмі нижче.

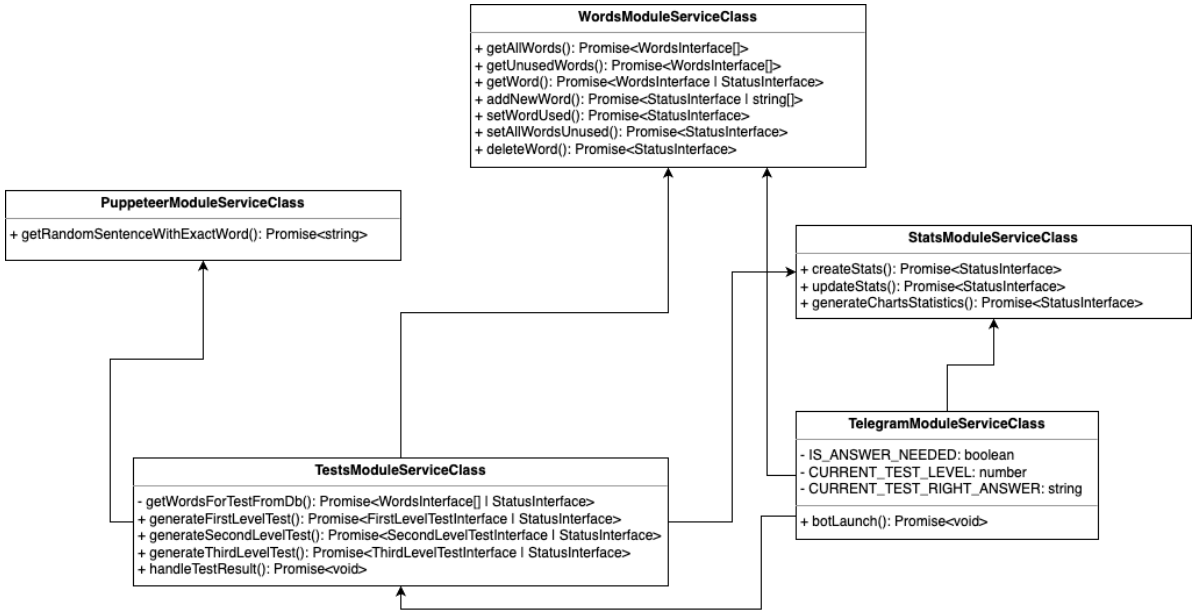

Рисунок 3.2 – Діаграма класів

В наведеній архітектурі TelegramModuleServiceClass виступає у ролі рівню представлення, у цьому класі описана логіка взаємодії користувача з додатком, тому він залежить від трьох основних класів.

PuppeteerModuleServiceClass – клас, який допомагає TestsModuleServiceClass з генерацією тесту третього рівня. З допомогою пакета puppeteer логіка методу цього сервісу забирає список речень з якимось словом зі словника з сайту [1] та вибирає одне випадкове.

Два останні класи відповідальні за маніпуляції зі статистикою та словами. Також StatsModuleServiceClass має метод для генерації графіків статистики проходження тестів на основі даних з таблиці stats у базі даних.

3.3.2 Діаграма послідовності

Опишемо структуру послідовності користування застосунком користувачем на рисунку 3.2. У цій конкретній ситуації відтворено послідовність проходження тесту третього рівня користувачем.

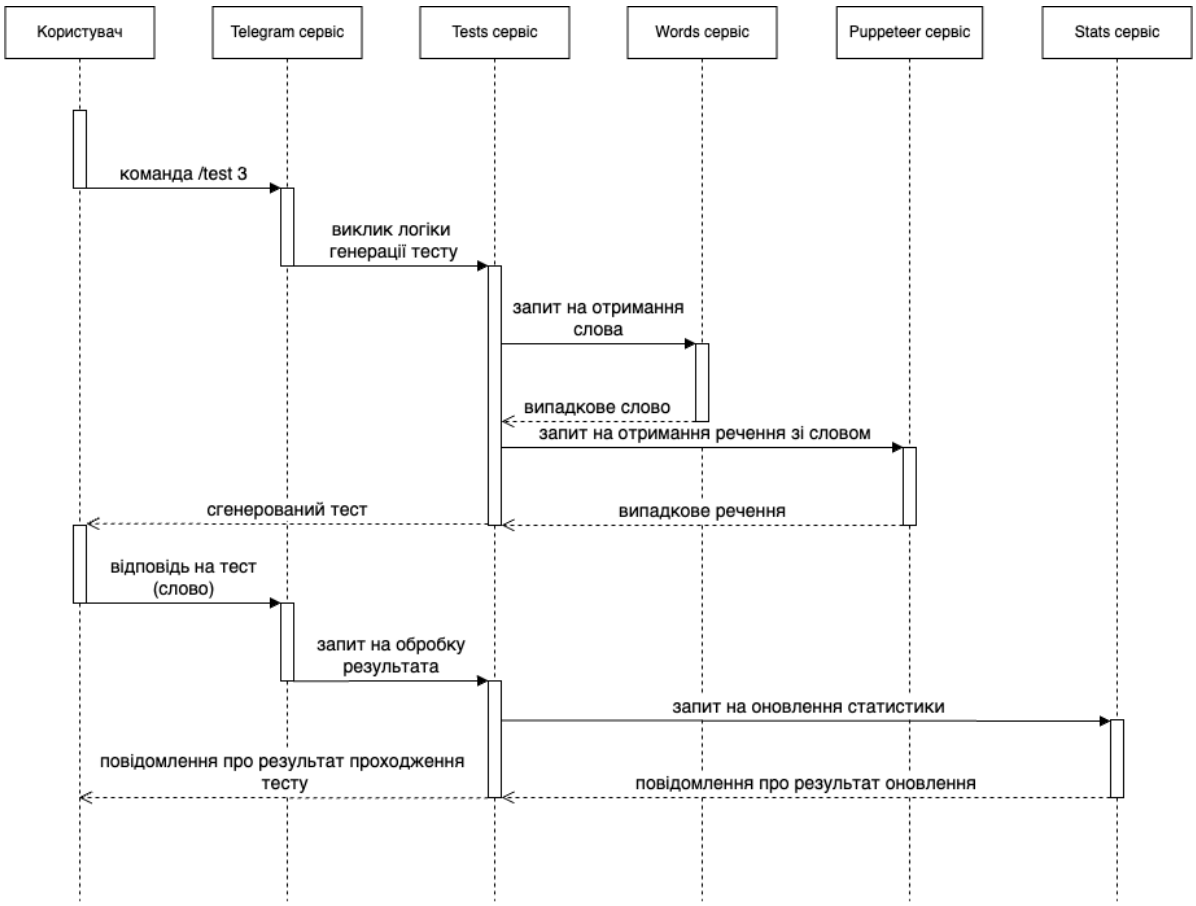

Рисунок 3.3 – Діаграма послідовності проходження третього тесту

Увесь процес проходження користувачем тесту третього рівня складається з наступних кроків:

– користувач вводить команду /test з параметром 3, після чого Telegram сервіс проводить валідацію введених даних та викликає Tests сервіс та запускає процес генерації тесту;

– після початку процесу Tests сервіс забирає випадкове невикористане слово з бази даних та помічає це слово як використане. Ці операції робить Word сервіс, який викликається Tests сервісом;

– далі Test сервіс звертається до Puppeteer сервісу, для отримання випадкового речення з необхідним словом за допомогою бібліотеки puppeteer, після чого генерує тест та відправляє користувачу;

– після отримання готового тесту користувач вводить відповідь, Telegram модуль її валідує, та викликає Test сервіс, для обробки результату тесту;

– для збереження результату Test сервіс звертається до Stats сервісу, оновлюючи статистику в базі, після чого надсилає користувачу повідомлення про результат проходження тесту.

3.3.3 Діаграма компонентів

Основна логіка системи міститься у наступних компонентах:

– PuppeteerModule – модуль відповідальний за всю логіку роботи з пакетом puppetier. Він здійснює отримання випадкового речення з необхідним словом. Використовується в модулі TestsModule для отримання речення для тесту третього рівня;

– WordsModule – модуль відповідальний за взаємодію з таблицею words. Містить методи для отримання списку всіх збережених слів та додавання нового слова. Використовується в модулі TestsModule для отримання слова для тесту, та в модулі TelegramModule для надання можливості маніпуляцій зі словами користувачу безпосередньо;

– StatsModule – модуль відповідальний за взаємодією з таблицею stats та генерацією графіків статистики. Створює запис у таблиці кожну добу, містить логіку змінення останнього запису в залежності від пройдених за добу тестів та логіку створення графіків статистики. Використовується в модулі TestsModule для змінення останнього запису після проходження кожного тесту, та в модулі TelegramModule для надання можливості перегляду графіків користувачем;

– TestsModule – основний модуль бота для створення тестів та обробку відповідей на них. Містить методи створення тестів усіх рівнів та логіку перевірки результату і оновлення статистики (викликаючи метод StatsModule). Використовується в модуль TelegramModule, для надання користувачу можливості проходити тести;

– TelegramModule – цей модуль сбирає в собі логіку всіх інших, викликаючи їх методи через команди бота.

Нижче представлена діаграма компонентів на рисунку 3.4.

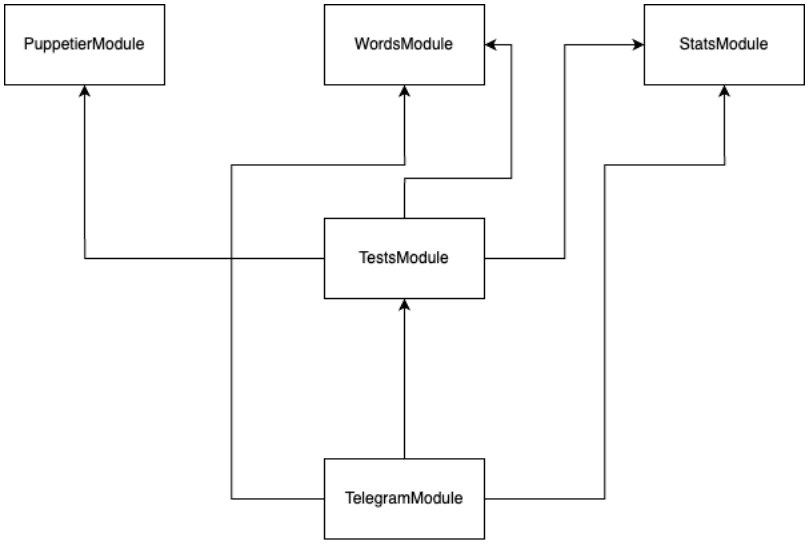

Рисунок 3.4 – Діаграма компонентів

Ці компоненти є основними, також додаток містить більш окремі функції та допоміжні класи для функціонування основних модулів.

3.3.4 Специфікація функцій

Згідно до діаграм класів, наведеній у розділі 3.3.1, буде наведено опис основних методів класів.

Таблиця 3.1 – Опис методів у класі WordsModuleServiceClass

Назва метода	Опис метода
getAllWords	Метод для отримання усіх слів з бази даних
getUnusedWords	Метод для отримання усіх невикористаних слів з бази даних (поле is_used_per_day – false)

getWord	Метод для отримання конкретного слова з бази за його англійською версією
addNewWord	Метод для додання нового слова до бази, очікує версію на англійській, автоматично перекладає її на українську та зберігає в базі
setWordUsed	Метод, який помічає конкретне слово як використане (поле is_used_per_day – true)
setAllWordsUnused	Метод, який помічає усі слова в базі як невикористані (викликається, коли в базі не залишилось невикористаних слів)
deleteWord	Метод для видалення конкретного слова з бази за його англійською версією

Таблиця 3.2 – Опис методів у класі StatsModuleServiceClass

Назва метода	Опис метода
createStats	Метод для створення статистики в базі (викликається раз на добу завдяки node-cron)
updateStats	Метод для оновлення поточної статистики після проходження користувачем тесту

generateChartsStatistics	Метод для генерації графіків статистки. Перший графік – кількість зароблених користувачем балів по днях. Другий графік – співвідношення правильних та помилкових відповідей по днях

Таблиця 3.3 – Опис методів у класі PuppeteerModuleServiceClass

Назва метода	Опис метода
getRandomSentenceWithExactWord	Метод, який отримує список речень з необхідним словом з сайту [1] за допомогою бібліотеки puppeteer та

Таблиця 3.4 – Опис методів у класі TestsModuleServiceClass

Назва методу	Опис методу
getWordsFromTestFromDb	Метод, який отримує слово або слова для тесту, у випадку, коли невикористаних слів немає, викликає метод класу WordsServiceClassModule setAllWordsUnused
generateFirstLevelTest	Метод для генерації тесту першого рівня, повертає питання, та 3 варіанти відповіді
generateSecondLevelTest	Метод для генерації тесту другого рівня, повертає питання

generateThirdLevelTest	Метод для генерації тесту третього рівня, викликає метод getRandomSentenceWithExactWord класу PuppeteerModuleServiceClass, повертає речення, з заміненим словом на пропуск
handleTestResult	Метод для обробки результату тесту, оновлює статистику викликаючи метод updateStats класу StatsModuleServiceClass (зберігає бал, що відповідає рівню теста)

Таблиця 3.5 – Опис методів у класі TelegramModuleServiceClass

Назва методу	Опис методу
botLaunch	Метод, який запускає телеграм бота, оброблює різні сценарії залежно від введених користувачем даних, проводить валідацію

Висновки до розділу

У даному розділі були повністю описані технології обрані для розробки даного програмного рішення. Також була надана інформація про оптимальне технічне забезпечення, для нормального функціонування системи Була детально описана та охарактеризована архітектура додатку.

Була розглянута діаграма послідовності взаємодії користувача з додатком. Була представлена структура основних класів та їх зав’язків. Також була надана уся інформація щодо методів кожного з основних класів, та їх функціонування.

Створення коректної архітектури додатку є важливим аспектом розробки програмного забезпечення і має значний вплив на його якість, розширюваність та підтримку. Ось кілька причин, чому коректна архітектура є важливою:

– розширюваність: Коректна архітектура дозволяє легко розширювати функціональність додатку з мінімальними змінами в існуючому коді. Це робить процес розробки швидшим і ефективнішим, особливо у випадку, коли потрібно внести зміни або додати нові функції в майбутньому;

– підтримка та розуміння коду: Хороша архітектура дозволяє зрозуміло організувати код, розділяючи його на логічні компоненти та шари. Це сприяє легкому розумінню та збереженню кодової бази під час розвитку, підтримки та співпраці з іншими розробниками;

– тестування та налагодження: Коректна архітектура сприяє легкості тестування і налагодження додатку. Вона дозволяє використовувати модульні тести та інші методи тестування для перевірки окремих компонентів без необхідності запускати всю систему. Це полегшує виявлення та виправлення помилок;

– швидкодія та масштабованість: Грамотно спроектована архітектура дозволяє досягти оптимальної швидкодії та ефективності роботи додатку. Вона допомагає уникнути зайвого навантаження, забезпечує ефективне використання ресурсів та легко масштабується для обробки зростаючого обсягу даних або навантаження;

– підтримка розробників: Хороша архітектура спрощує співпрацю розробників та сприяє швидкому вступу нових команд у проект. Вона створює чіткі інтерфейси та документацію, які полегшують розуміння системи та спілкування між командами розробників.

Враховуючи ці фактори, важливо приділити належну увагу процесу проектування архітектури додатку, ретельно вивчити вимоги, забезпечити гнучкість та розширюваність системи та забезпечити ефективну роботу додатку як у поточному, так і в майбутньому.

4 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

4.1 Керівництво користувача

Розглянемо функціонал, який надає користувачу даний застосунок. Перше, що має право зробити користувач – запустити бота і отримати привітальне повідомлення, що продемонстровано на рисунку 4.1.

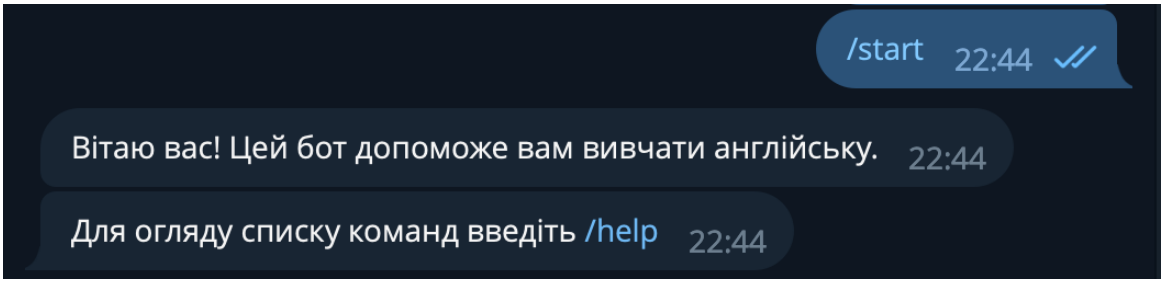

Рисунок 4.1 – Запуск бота користувачем

Наступним кроком користувач може отримати список доступних команд, для цього йому потрібно ввести команду /help, як показано на рисунку 4.2.

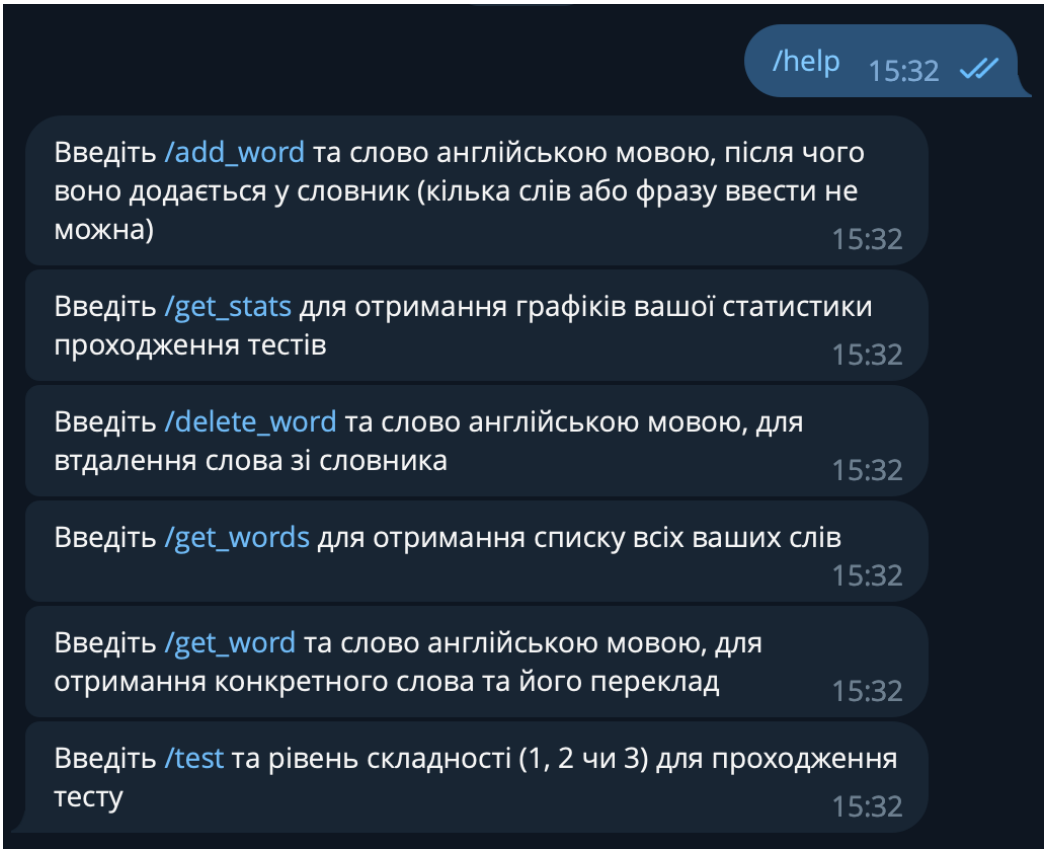

Рисунок 4.2 – Отримання списку команд

Одна з головних функцій, яку надає бот – це додавання слова. Для цього користувачу потрібно ввести команду /add_word та слово англійською мовою, як показано на рисунку 4.3. Вводити більше одного слова функціоналом не передбачено, тож користувач отримає повідомлення про помилку.

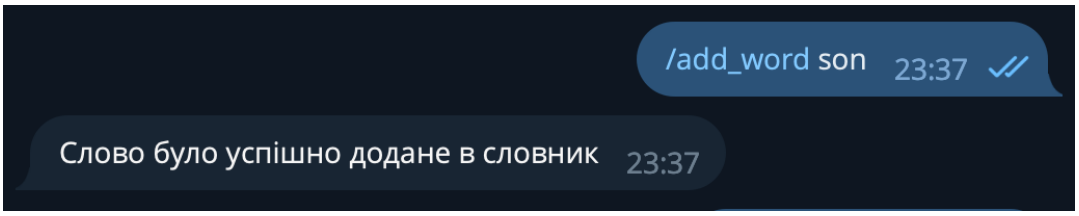

Рисунок 4.3 – Додання слова у словник

Якщо користувач ввів слово з помилкою – додаток запропонує йому слова, які схожі на введене слово, це демонструється на рисунку 4.4.

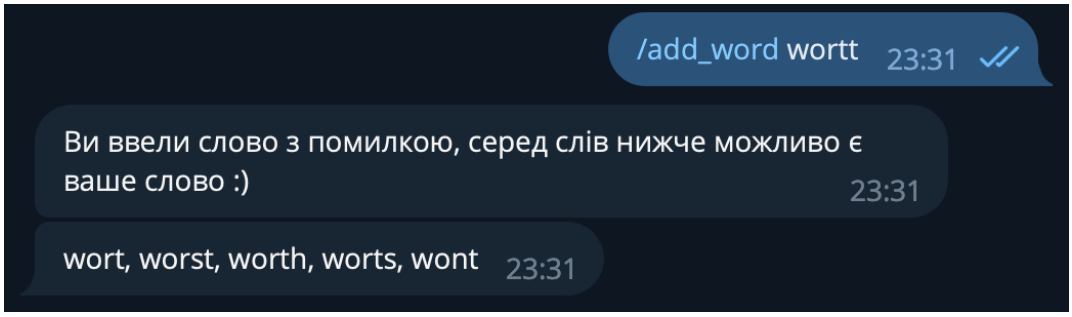

Рисунок 4.5 – Список запропонованих слів, якщо введене слово невірне

Користувач має можливість отримати список слів зі свого словника через команду /get_words, як показано на рисунку 4.5.

Рисунок 4.5 – Отримання списку слів

Також, можна отримати одне конкретне слово за його англійською версією, як на рисунку 4.6. Застосунок поверне повідомлення про помилку, якщо введеного слова нема в словнику.

Рисунок 4.6 – Отримання конкретного слова за його англійською версією

Фінальною функцією маніпуляцій зі словами є видалення слова. Для цього користувач може ввести команду /delete та слово англійською, як показано на рисунку 4.7. Якщо такого слова нема в базі, повернеться повідомлення про помилку.

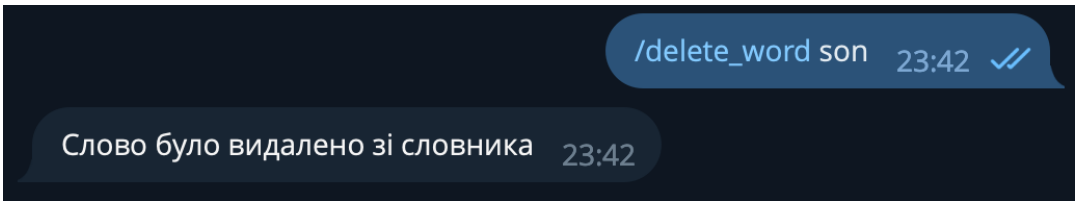

Рисунок 4.7 – Видалення слова зі словника

Для того, щоб отримати графіки статистики проходження тестів, користувач може ввести команду /get_stats. На рисунку 4.8 показано отримання двох графіків прогресу: перший графік – кількість зароблених користувачем балів по днях, другий графік – співвідношення правильних та помилкових відповідей по днях

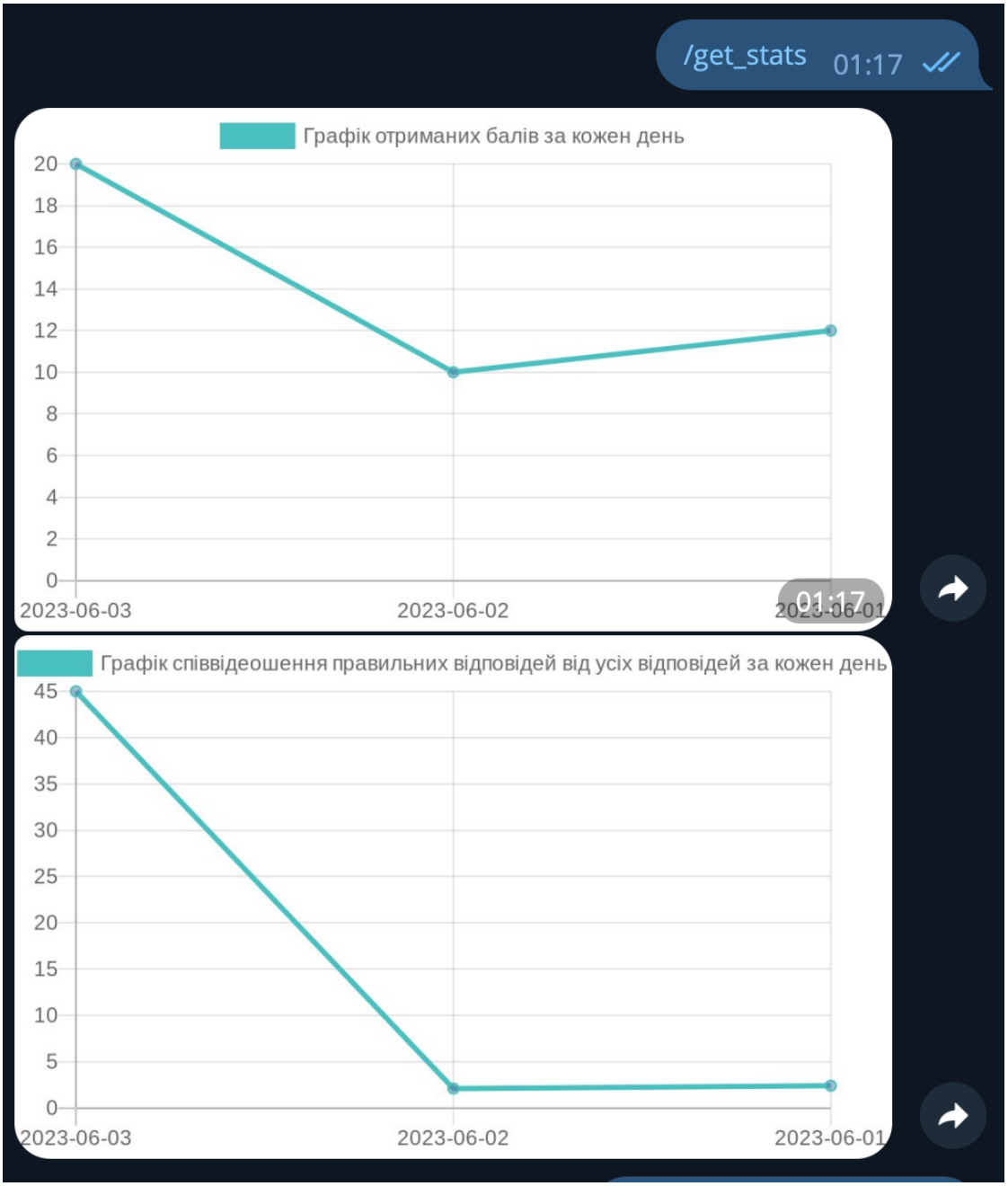

Рисунок 4.8 – Отримання графіків статистики проходження тестів

Користувач має можливість проходження тестів різного рівню, усього таких рівня три.

Перший рівень тесту пропонує користувачу перевести одне зі слів за словника з української на англійську. На вибір у користувача є 3 варіанти відповіді, це показано на рисунку 4.9. За цей тест нараховується один бал.

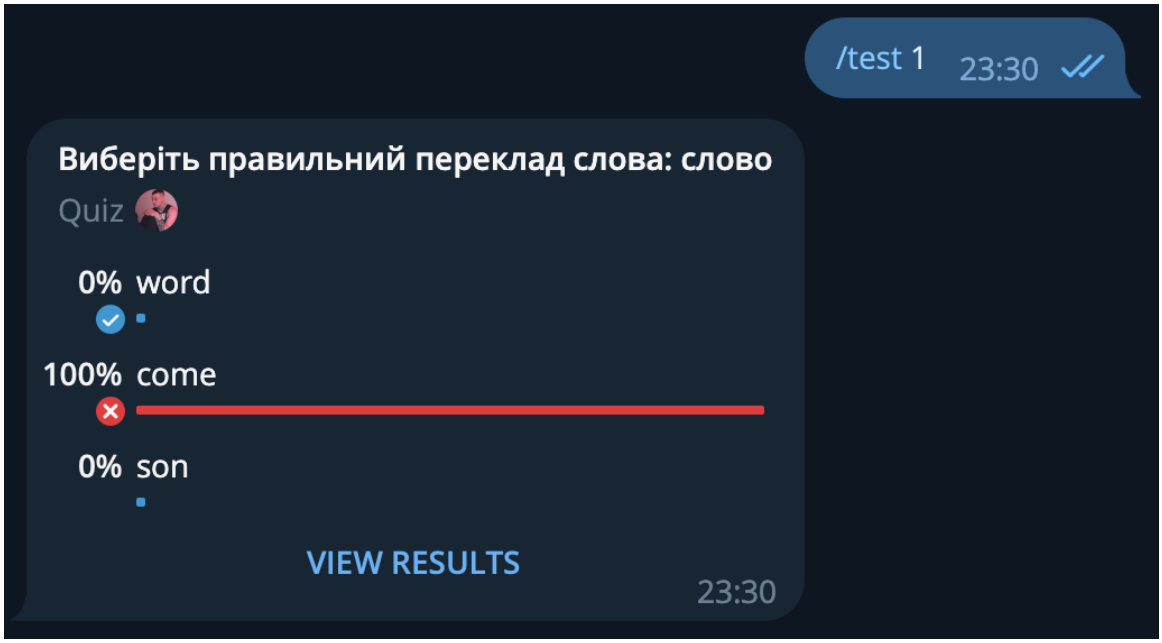

Рисунок 4.9 – Проходження тесту першого рівня.

На другому рівні тест також пропонує користувачу перевести одне зі слів за словника з української на англійську, але користувач повинен самостійно ввести свою відповідь, це демонструється на рисунку 4.10. За цей тест нараховуються два бали.

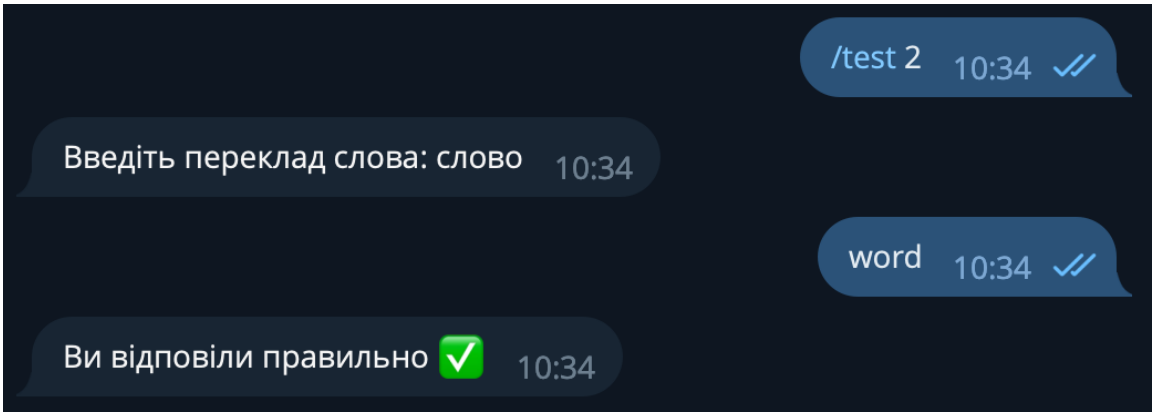

Рисунок 4.10 – Проходження тесту третього рівня

Тест третього рівня надає користувачу випадкове речення, в якому замість конкретного слова стоїть пропуск, користувач повинен ввести це слово в якості відповіді, це показано на рисунку 4.11. За цей тест нараховуються три бали.

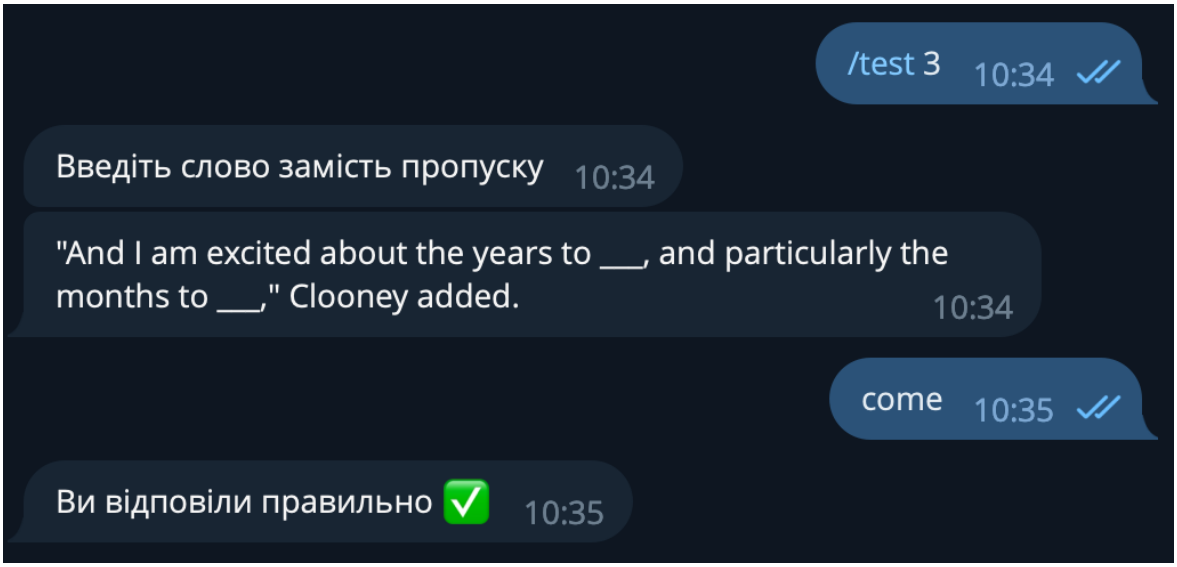

Рисунок 4.11 – Проходження тесту третього рівня

4.2 Випробування програмного продукту

4.2.2 Мета випробувань

Метою випробувань є перевірка коректності роботи всього наданого функціоналу системи. Необхідна перевірка процесу введення та збереження даних, генерації тестів та графіків статистики.

Для досягнення поставленої мети необхідно визначити необхідний набір сценаріїв для тесту, які охоплюють розроблений функціонал.

Згідно з розробленими сценаріями необхідно виконати кожен з них, занотувати початковий стан системи, вхідні дані. Описати конкретний план проведення тесту, навести очікуваний результат, та порівняти його з результатом, що видала система. Потім перевірити стан системи та цілісність її даних.

4.3.3 Загальні положення

Для забезпечення повного циклу тестування був обраний наступний комплекс тестувань – функціональне тестування, тестування з навантаженням, регресійне тестування.

Функціональне тестування - це процес перевірки, чи відповідає програмне забезпечення вимогам та очікуванням функціональності. Основна мета функціонального тестування полягає у перевірці, чи працює програма правильно, виконує необхідні функції та відповідає очікуваному поведінці.

У процесі функціонального тестування перевіряються різні аспекти програми, такі як вхідні та вихідні дані, функціональність окремих функцій, поведінка програми в різних сценаріях використання тощо. Основні критерії функціонального тестування включають правильність результатів, коректність обробки вхідних даних, відповідність програми специфікаціям та вимогам, забезпечення необхідної функціональності.

Функціональне тестування може виконуватись ручним чином або автоматизованими засобами. Використання автоматизованих інструментів дозволяє прискорити процес тестування, забезпечити повторюваність тестових сценаріїв та швидше виявляти помилки.

Підхід до функціонального тестування може варіюватися в залежності від конкретних потреб проекту. Важливо визначити масштаб тестування, скласти тестові сценарії, виконати їх і проаналізувати результати для виявлення потенційних проблем та дефектів.

Функціональне тестування є важливою складовою процесу розробки програмного забезпечення, оскільки допомагає впевнитись, що програма виконує очікувану функціональність та забезпечує задану якість. Цей тип тестування сприяє поліпшенню надійності та стабільності програмного забезпечення, а також забезпечує задоволення потреб користувачів.

Тестування з навантаженням (load testing) є процесом перевірки реакції програмного забезпечення на певне навантаження або обсяг вхідних даних. Його основна мета полягає в оцінці та перевірці працездатності, стабільності та продуктивності системи за умови навантаження, яке наближене до реальних умов експлуатації.

Під час тестування з навантаженням створюються сценарії, які симулюють одночасне використання системи багатьма користувачами або генерують значний

обсяг запитів до системи. Таким чином, перевіряються такі аспекти, як швидкодія системи, її відповідь на навантаження, масштабованість та здатність до обробки багатьох запитів одночасно.

Тестування з навантаженням може включати такі типи тестів:

1. Тестування продуктивності: оцінка часу відповіді системи, швидкодії виконання операцій, пропускної здатності та ресурсоємності системи під час роботи зі збільшеним навантаженням.
2. Тестування стабільності: перевірка стійкості системи під тривалим навантаженням, виявлення можливих проблем зі скиданням пам'яті, витоками ресурсів або відмовами під великим навантаженням.
3. Тестування масштабованості: перевірка здатності системи збільшувати обсяг обробки даних або кількість одночасних користувачів шляхом збільшення обладнання або розширення інфраструктури.
4. Тестування навантаження реального середовища: виконання тестів на реальному середовищі з використанням даних, що наближені до реальних сценаріїв використання, для перевірки працездатності та надійності системи.

В результаті проведення тестування з навантаженням можна отримати інформацію про продуктивність, стійкість та масштабованість системи, ідентифікувати можливі проблеми та здатність системи впоратися з великим обсягом роботи. Це дозволяє розробникам та тестувальникам вдосконалювати систему, забезпечуючи її оптимальну працездатність та задоволення потреб користувачів.

Регресійне тестування є процесом перевірки вже раніше протестованих функцій або модулів програмного забезпечення з метою переконатися, що внесені зміни або додатковий функціонал не спричинили появу нових помилок або проблем у вже працездатних частинах системи.

Головна мета регресійного тестування полягає в забезпеченні стабільності та надійності програмного забезпечення після внесення змін. Це важливо, оскільки при розробці програмного продукту часто вносяться зміни у вихідний код, додаються нові функції або виправляються помилки. Проте, ці зміни можуть мати

непередбачувані впливи на існуючий функціонал, і регресійне тестування допомагає виявити такі проблеми.

Під час регресійного тестування повторно виконуються вже пройдені тестові сценарії, тестові набори даних або автоматизовані тести з метою перевірки, чи продукт після внесених змін працює правильно та не має нових дефектів або помилок. Також можуть бути використані спеціальні інструменти для автоматичного виконання регресійних тестів та порівняння результатів з попередніми версіями.

Регресійне тестування важливо для забезпечення якості програмного продукту, оскільки воно дозволяє виявляти можливі проблеми та вади, які можуть з'явитися після внесення змін. Це допомагає підтримувати стабільність системи та забезпечувати безперебійну роботу програмного забезпечення після кожного випуску нової версії або внесення змін.

4.2.3 Результати випробувань

Згідно вибраної методики тестувань були створені тест-кейси, які покривають наданий у додатку функціонал.

Таблиця 4.1 – Тестування додання слова

Мета тесту	Перевірка додання нового слова
Початковий стан системи	Слова в базі не існує
Вхідні дані	Команда /add_word та коректне введене слово англійською мовою
Схема проведення тесту	Користувач вводить команду /add_word та одне коректне слово англійською мовою
Очікуваний результат	Слово було переведене на українську та був створений новий запис у

	таблиці words. Користувач отримав повідомлення про успішне додання слова

Таблиця 4.2 – Тестування некоректного додання слова

Мета тесту	Перевірка додання нового слова
Початковий стан системи	Слова в базі не існує
Вхідні дані	Команда /add_word та декілька слів англійською мовою
Схема проведення тесту	Користувач вводить команду /add_word та декілька слів чи фразу англійською мовою
Очікуваний результат	Користувач отримав повідомлення про помилку. Запису в базі не з’явилось

Таблиця 4.3 – Тестування отримання списку слів

Мета тесту	Перевірка отримання списку слів
Початковий стан системи	Слова існують в базі
Вхідні дані	Команда /get_words
Схема проведення тесту	Користувач вводить команду /get_words
Очікуваний результат	Користувач отримав список слів в базі

Таблиця 4.4 – Отримамння слова з бази

Мета тесту	Перевірка отримання слова з бази
Початковий стан системи	Слово існує в базі
Вхідні дані	Команда /get_word та слово англійсською
Схема проведення тесту	Користувач вводить команду /get_word та одне слово англійською
Очікуваний результат	Користувач отримав слово з бази

Таблиця 4.5 – Тестування некоректного отримання слова

Мета тесту	Перевірка некоректного отримання слова
Початковий стан системи	Слово існує в базі
Вхідні дані	Команда /get_word та слово англійсською
Схема проведення тесту	Користувач вводить команду /get_word та некоректно введене слово англійською
Очікуваний результат	Користувач отримав повідомлення про помилку

Таблиця 4.6 – Тестування видалення слова

Мета тесту	Перевірка видалення слова
Початковий стан системи	Слово існує в базі

Вхідні дані	Команда /delete_word та слово англійською
Схема проведення тесту	Користувач вводить команду /delete_word та слово англійською
Очікуваний результат	Користувач отримав повідомлення про успішне видалення слова. Запис з бази видаляється

Таблиця 4.7 – Тестування некоректного видалення слова

Мета тесту	Перевірка некоректного видалення слова
Початковий стан системи	Слово існує в базі
Вхідні дані	Команда /delete_word та слово англійською
Схема проведення тесту	Користувач вводить команду /delete_word та некоректно введене слово англійською
Очікуваний результат	Користувач отримав повідомлення про помилку. Запис у базі залишається

Таблиця 4.8 – Тестування отримання графіків статистики

Мета тесту	Перевірка отримання графіків статистики
Початковий стан системи	Записи статистики існують в базі

Вхідні дані	Команда /get_stats
Схема проведення тесту	Користувач вводить команду /get_stats
Очікуваний результат	Користувач отримує два зображення графіків статистики

Таблиця 4.9 – Тестування генерації тесту

Мета тесту	Перевірка генерації тесту
Початковий стан системи	Записи статистики та слів існують в базі
Вхідні дані	Команда /test та рівень тесту
Схема проведення тесту	Користувач вводить команду /test та рівень тесту
Очікуваний результат	Користувач отримує згенерований тест конкретного рівня. Використане в тесті слово з бази помічається як використане

Таблиця 4.10 – Тестування обробки результату проходження тесту

Мета тесту	Перевірка генерації тесту
Початковий стан системи	Запис поточної статистики існує в базі
Вхідні дані	Рівень тесту, значення результату тесту (true чи false)

Схема проведення тесту	Користувач вводить відповідь до тесту
Очікуваний результат	Користувач отримує повідомлення про результат проходження тесту, поточний запис статистики в базі оновлюється

Висновки до розділу

В даному розділі було надано керівництво по усьому функціоналу застосунка користувачу. Був сформульований комплекс тестувань для перевірки коректної роботи застосунку. Були описані тест-кейси та очікувані результати тестувань.

Система показала очікувані результати, тому можна зробити висновок, що вона відповідає визначеним функціональним вимогам.

Тестування додатку є невід'ємною частиною розробки програмного забезпечення і має велику важливість з кількох причин:

– виявлення помилок: Тестування допомагає виявляти помилки та дефекти в додатку ще до його випуску. Це дозволяє виправити проблеми та забезпечити високу якість продукту. Виявлення помилок на ранніх етапах розробки також зменшує вартість виправлення помилок у майбутньому;

– забезпечення функціональності: Тестування переконується, що додаток відповідає вимогам та виконує задану функціональність. Це гарантує, що користувачі отримують очікувані результати та можуть використовувати додаток без проблем;

– підтримка якості: Тестування допомагає забезпечити високу якість додатку шляхом перевірки правильності роботи функцій, відповідності стандартам і вимогам, а також виявлення та усунення вразливостей безпеки. Це важливо для задоволення потреб користувачів і підтримки доброї репутації додатку;

– підтримка розширюваності: Регулярне тестування дозволяє впевнитися, що нові функції або зміни в додатку не порушують роботу вже існуючих функцій. Це допомагає забезпечити, що додаток легко розширюється і масштабується без негативного впливу на його функціональність;

– довіра користувачів: Регулярне тестування додатку допомагає зберегти довіру користувачів. Коли користувачі знають, що додаток перевірений та протестований, вони відчувають більшу впевненість в його надійності та використанні.

Всі ці фактори роблять тестування незамінною складовою процесу розробки додатку, допомагаючи забезпечити якість, надійність та задоволення користувачів.

ВИСНОВКИ

Під час створення застосунку для вивчення мови, треба визначити основні його елементи та функції, які зроблять процес вивчення цікавим та ефективним для користувача. Можливість мати власний словник зробить додаток гнучким, і користувач буде мати можливість самостійно визначати релевантність тих чи інших слів. Але просто мати список слів – не дуже ефективно, застосунок повинен впроваджувати інтерактив. Для цього ідеально підходять тести, користувач буде мати можливість перевірити знання слів з власного словника. Також важливим елементом є можливість слідкувати за прогресом, для цього підходять графіки статистики проходження тестів. Дана програмна реалізація містить в собі всі вище перераховані аспекти.

Головною технічною задачею є створення телеграм бота, для вивчення англійської мови, з функціями маніпуляцій з власним словником, проходженням тестів зі доданими словами та переглядом статистики прогресу.

На першому етапі було спроектоване рішення додатку для вивчення англійської. Були описані предметне середовище та процес діяльності. Був описаний функціонал застосунку. Були проаналізовані аналогічні додатки, та проведено їх порівняння. Була визначена мета продукту та задачі для її досягнення.

Другим етапом було створення структури бази даних. Після визначення основних цілей додатку були сформульовані основні сутності, які треба зберігати в базі даних. Також був наданий опис полів створених таблиць та її призначення.

Третій етап – створення архітектури додатку. Були розглянуті технології, які знайшли своє застосування під час розробки системи. Була детально описана та охарактеризована архітектура додатку, спроектована діаграма послідовності та структури класів. Також була надана інформація про оптимальне технічне забезпечення, для нормального функціонування системи.

Завершальним етапом було надано керівництво по усьому функціоналу застосунка користувачу. Був сформульований комплекс тестувань для перевірки коректної роботи застосунку. Були описані тест-кейси та очікувані результати тестувань, після чого увесь наданий функціонал системи був протестований.

Підсумовуючи пророблену роботу, можна сказати, що усі поставлені цілі з розробки застосунку для допомоги вивчення англійської мови були досягнуті. З імплементацією перерахованих технологій був відтворений зазначений функціонал. Також, цей функціонал був протестований, з чого можна зробити висновок, що система функціонує правильно.

ПЕРЕЛІК ПОСИЛАНЬ

1. Ресурс для отримання випадкових речень з конкретним словом: веб-сайт. URL: https://www.randomsentencegen.com/ (дата звернення 20.04.2023)
2. Документація NodeJs: веб-сайт. URL: https://nodejs.org/en (дата звернення 15.04.2023)
3. Документація TypeScript: веб-сайт. URL: https://www.typescriptlang.org/ (дата звернення 16.04.2023)
4. Документація бази даних PostgreSQL: веб-сайт. URL: (https://telegraf.js.org/ дата звернення 16.04.2023)
5. Документація інструменту Telegraf: веб-сайт. URL: https://www.npmjs.com/package/telegraf (дата звернення 13.05.2023)
6. Документація елементу Puppeteer: веб-сайт. URL: https://pptr.dev/ (дата звернення 20.04.2023)
7. Документація інструменту для перекладу слів: веб-сайт. URL: https://snyk.io/advisor/npm-package/@iamtraction/google-translate (дата звернення 22.04.2023)
8. Мартін Роберт. Чиста Архітектура: Мистецтво створення програмного забезпечення. Видання друге / пер. з англ. І. Бондар-Терещенко. - Харків : Вид-во «Ранок» : Фабула, 2023. - 368 с.
9. Архітектурні паттерни та їх використання: веб-сайт. URL: https://refactoring.guru/uk/design-patterns/strategy (дата звернення 22.04.2023)
10. Документація методів та класів JavaScript: веб-сайт. URL: https://developer.mozilla.org/ru/docs/Web/JavaScript (дата звернення 17.04.2023)
11. Telegram bot API документація: веб-сайт. URL: https://core.telegram.org/bots/api (дата звернення 21.05.2023)
12. Документація node-cron: веб-сайт. URL: https://www.npmjs.com/package/node-cron (дата звернення 30.05.2023)

13. Вихідний код бібліотеки typo-js: веб-сайт. URL: https://github.com/cfinke/Typo.js (дата звернення 19.04.2023)
14. Документація інструменту для взаємодії з базою даних pg: веб-сайт. URL: https://node-postgres.com/ (дата звернення 16.04.2023)
15. Відкритий код бібліотеки telegraf: веб-сайт. URL: https://github.com/telegraf/telegraf (дата звернення 15.05.2023)
16. Приклад функціонуючого телеграм бота на Node.Js: веб-сайт: URL: https://medium.com/geekculture/build-a-telegram-bot-using-typescript-node-js-and-telegraf-and-deploy-it-on-heroku-fcc28c15614f (дата звернення 20.04.2023)